Borland®
(A MICRO FOCUS COMPANY)

MICRO FOCUS®

SilkCentral Test Manager 12.0

Help

Micro Focus
575 Anton Blvd., Suite 510
Costa Mesa, CA 92626

Copyright © 2012 Micro Focus. All rights reserved. Portions Copyright © 2004-2009 Borland Software Corporation (a Micro Focus company).

MICRO FOCUS, the Micro Focus logo, and Micro Focus product names are trademarks or registered trademarks of Micro Focus IP Development Limited or its subsidiaries or affiliated companies in the United States, United Kingdom, and other countries.

BORLAND, the Borland logo, and Borland product names are trademarks or registered trademarks of Borland Software Corporation or its subsidiaries or affiliated companies in the United States, United Kingdom, and other countries.

All other marks are the property of their respective owners.

2012-03-08

Contents

SilkCentral Test Manager 12.0 6

Getting Started 6

Tour of the UI 6

What's New in SilkCentral Test Manager 12.0 9

Installing and Licensing Test Manager 12

SilkCentral Issue Manager 12

Browser Settings 12

Logging In to and Out from Test Manager 12

Working with SilkPerformer Projects 13

Working with Silk Performance Explorer 13

SilkCentral Administration Module 13

Test Manager, Risk-Based Testing, and Quality Goals 15

Quick Start Tasks 17

My Dashboard 41

Adding Panels to the Dashboard 41

Dashboard Panels 42

Dashboard Panel Permissions 43

Projects 44

Build Information 44

Comparing a Project with a Baseline 45

Selecting Projects 45

Project List 46

Switching to a Recently-Accessed Project 46

Settings Configuration 46

Requirements 100

Managing Requirements 100

Requirements Toolbar Functions 102

Requirements Tree 104

Requirements Document View 104

Requirement Properties 105

Requirement Attachments 107

Working with Tests 109

Coverage 112

Requirement History 114

Requirement Change Notification 115

Requirements Reports 115

Flags 118

Requirements Import 119

External Requirements Management Tools 122

Tests 134

Tests Document View 134

Working with the Tests Tree 134

Test Toolbar Functions 139

Test Reports 141

Success Conditions 144

Test Containers 145

Test Folders 147

Test Packages 148

Test History 151

Upload Manager 152

Tests 155

Libraries 186
Windows Script Host Tests 194
Test Export-Update and Import 198
Integrating Test Manager into Rally 203
Execution Planning 205
Execution Planning Toolbar Functions 205
Manual Execution Planning 206
Configuration Testing 219
Execution Planning Document View 224
Working with the Execution Plans Tree 225
Executing Tests Assigned to Execution Plans 250
Analyzing Test Runs 269
Current Run Page 276
Working with SilkPerformer Projects 279
SilkTest Classic Tests 283
VMware Lab Manager Integration 284
Tracking 284
Activities 284
Project Overview Report 293
Quality Goals 293
Analyzing Results 295
Video Capturing 295
Screen Capturing 295
Issues 296
Creating New Issues 296
Viewing Issue Statistics in Document View 297
Issues Document View 297
Viewing Issue Statistics in Details View 297
Issues Page 298
Assigning External Issues 298
Synchronizing Internal and External Issue States 298
Deleting Issues 298
Specifying a Calendar Range 299
Calendar Tool 299
Integrations 299
Code Analysis Tools Integration 299
Issue Tracking Profile Integrations 300
Other Integrations 312
Requirement Management Tools 315
Source Control Profile Integrations 322
Test Automation Tools 339
Communicating with an External System Over SSL 360
Reports 361
Default Reports 361
Creating and Customizing Reports 383
Reporting Area 390
Code Analysis 396
Filtering 419
Filters 419
Global Filter Details 419
Recent Changes 420
Creating Filters 420
Creating Advanced Filters 422
Applying Filters 422
Editing Filters 422
Copying Filters 423

Deleting Filters 423
System Diagnostics 423
Viewing System Diagnostics 424

SilkCentral Test Manager 12.0

SilkCentral® Test Manager™ (Test Manager) is a powerful, all-inclusive, software test-management tool. Test Manager builds quality and productivity into your testing process, speeding the delivery of successful software projects while minimizing the risks of application failures. It consolidates all the critical software-testing phases within a single, scalable, Web-based testing system, enabling your local and distributed software-development teams to share experiences, resources, and critical information.

Getting Started

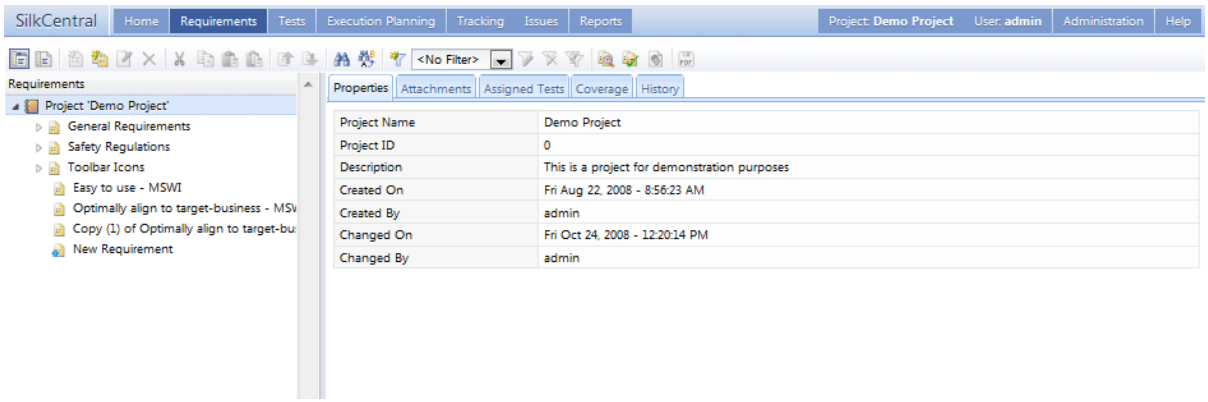
Test Manager promotes product quality throughout the development cycle.

Today's e-business systems are increasingly complex, and reliability is more important than ever. Therefore, assuring product quality throughout the development cycle is an important key to success. The best way to check a product's quality over time is to perform key tests on a daily basis. Test Manager helps in this regard by automating test executions that follow freely configurable schedules, both during product development and after deployment. By providing reports with different levels of detail, checking the status of products in development is as straightforward as checking an HTML report in a Web browser.

With complex software projects, thorough testing of new builds is critically important. Test Manager saves time and man hours by automating this process.

Tour of the UI

This topic provides an overview of the Test Manager user interface.



Basic UI Structure

The UI of Test Manager includes:

- The **Menu**: Placed across the top of the user interface. It gives you quick access to the functional units of Test Manager.
- The **Workspace**: In this example, the **Workspace** is the tabbed region showing the functional work area of the currently selected Test Manager unit. This view changes based on the unit you are working in.

The Units of Test Manager

The **Menu** gives you quick access to the functional units of Test Manager. It includes the following major sections:

Home

Displays the dashboard, which is your personal starting area. You can customize the dashboard by adding and arranging various panels, which give you the information you need for your daily work.

Requirements

Displays and enables you to maintain control over your project's requirements during development. You can perform the following tasks:

- Create, modify, and delete requirements.
- Associate tests with requirements.
- Track the change history of requirements.
- Generate test plans from requirement lists.

You can additionally access and manage the following:

- Properties of the requirements.
- Integrated external requirement-management tools.
- Requirement filters.
- Notifications that inform you about changes to the requirements.

Tests

Enables you to create and manage automated and manual tests. You can additionally access and manage the following:

- Libraries of shared steps, out of which commonly-used manual tests and test steps can be reused.
- Properties of the individual test steps for manual tests.
- Test attributes.
- The data sources that can be used for tests.
- Source control profiles, in which these tests are stored.
- Test filters.
- Notifications that inform you about changes to the tests.
- Version and build of the product that is tested with a specific test.

Execution Planning

Displays and allows you to manage the executions that execute the tests. You can perform the following tasks:

- Configure execution plans.
- Assign tests to execution plans.
- Set-up execution plan dependencies.
- Configure execution-server deployment, including the execution servers on which the tests are executed.
- Plan and manage the manual execution of tests.

You can additionally access and manage the following:

- Execution filters.
- Execution schedules.
- Version and build of the product that is tested with a specific execution of a test.
- Notifications that inform you about changes concerning the execution process.

Tracking

Displays the **Activities** page, the **Cross-Project Activities** page, the **Project Overview Report**, and **Quality Goals**.

The **Activities** page displays the activity for recently-executed, current, and upcoming execution plans on a per-project basis.

Issues

Displays and enables you to manage the issues that are related to the active project.

You can additionally access and manage the following:

- Projects in Issue Manager, when issues reside in Issue Manager.
- External issue tracking profiles, in which issues may possibly reside.

Reports

Displays and enables you to manage the reports which provide you with information on the requirements, tests, and execution plans. You can perform the following tasks:

- Generate reports.
- Download report templates.
- Edit report parameters.
- Create new reports based on pre-installed templates.

Additionally provides access to code analysis functionality information, with which you can evaluate the degree to which the code in your Application Under Test (AUT) is covered by test cases. You can then make informed estimates regarding effort, cost, and risk associated with specific code changes.

Projects

Displays and enables you to manage all projects in your Test Manager installation from a high-level test-manager's view. Additionally enables you to switch between projects, and manage project settings.

User

Displays and enables you to manage user settings. Additionally enables the following:

- Log out of Test Manager.
- Print the current Test Manager page.
- Bookmark the current Test Manager page. This is especially useful for bookmarking reports, where the current parameters are saved in the book-marked URL.

Administration

Displays and enables you to manage the administration settings. For additional information, refer to the *SilkCentral Administration Module Help*.

Help

Click to view context-sensitive help for the current page. Additionally provides access to the following:

- The Test Manager documentation.
- Tools that are shipped with Test Manager.
- About. Shows version and license information.

Context Menu Commands

Test Manager supports Windows-style context menus across many test management elements, for example tests, requirements, execution plans, folders, containers, reports, and more. Available through

right mouse-click, context menu commands typically include those commands that are available from each unit's toolbar. For elements listed in tree views, context menus offer commands for expanding and collapsing tree view elements. Commands that are not available to selected elements are disabled.

What's New in SilkCentral Test Manager 12.0

SilkCentral Test Manager 12.0 (Test Manager) introduces significant enhancements and changes.

Manual Execution Planning

The new manual execution planning approach assists you in selecting, distributing, and executing those manual tests that result in best achieving your quality goals under given time and resource constraints. Time and resources are managed together in the context of testing cycles. A *testing cycle* represents a period of time that has a set of manual testers and a set of tests assigned to it. Each manual tester in the testing cycle has a capacity, which is the amount of testing time that the tester has available for the testing cycle.

For test managers, the new **Manual Execution Planning** page provides the ability to efficiently plan the overall testing progress. The new page provides an answer to the often critical question "Are we able to finish our manual testing in time?", by providing answers to the following sub-questions:

- What manual tests are executed?
- When are those manual tests executed?
- Who executes those manual tests?
- Where are those manual tests executed?
- How far is the testing progress in the current testing cycle?

For manual testers, the new **Manual Execution Planning** page provides assistance in effectively prioritizing their work, displaying exactly which tests have to be done in a specific period of time.

The new manual execution planning approach interacts tightly with the new dashboard.

My Dashboard

The new dashboard in Test Manager is your personal starting area. You can customize it by adding and arranging various panels, which give you the information you need for your daily work.

You can add the following panels to the dashboard:

Panel	Description
Custom Information	This panel shows custom information for a project. Users can add news, descriptions or any other kind of relevant content to the panel.
Introduction	This panel gives you a quick introduction to Test Manager.
Issue Life Cycle	This panel shows the issues for a defined issue tracking profile and their status (open, fixed, verified, closed, deferred).
Issues Created per Tester	This panel shows the number of issues created by testers as a bar graph.
Manual Tests Assigned to Me	This panel shows all tests from a certain project that are assigned to you for execution. It enables you to quickly continue working on your tests and to get detailed information on all of your tests.
Quality Goal Progress	This panel shows the number of tests that must be executed to meet the quality goal in relation to the number of actual executed tests. The executed tests are divided into passed, failed, not executed, and N/A (not available).
Requirements Coverage Status	This panel shows the coverage of the requirements of the selected project broken down into passed, failed, not executed and not covered.

Panel	Description
Testing Cycle Progress	This panel shows a progress chart of passed, failed, in progress, and not yet executed tests for the defined testing cycle.
Testing Cycle Result Summary	This panel shows the status of the selected testing cycle and the progress status of the individual testers that are assigned to the testing cycle.
Testing Progress Across Testing Cycles	This panel shows the time in hours for the planned, in progress and completed tests. It helps a test manager to predict if the testing team will be able to finish the tests in time.
Volatile Tests	This panel shows how often tests changed their status within a defined time period. It helps identifying unstable tests.

System Diagnostics

The new *System Diagnostics* feature in **Help** > **About** enables you to quickly review and download relevant system settings and logs.

Calculated Property

A calculated property is used to create a single value based on a user-defined formula derived from the values of internal or custom properties. This value is then put into a labelled range called a *classification*. An example for a calculated property would be a *Risk* that is calculated based on two other properties like *Priority * Impact*. The result of this calculation for each requirement is mapped into classifications like *Low*, *Medium*, *High*. These classifications are then visible and usable in different locations throughout SilkCentral Test Manager.

Video Capturing in Web UI for Manual Testing

Manual testers using the Web UI for manual test execution can now capture execution videos. Such videos are then available as result files.

Assigning Issues to Manual Test Steps

This new feature allows you to assign issues directly to manual test steps. The created issues will contain the information about the steps they were created for as a link. Clicking this link takes the user directly to the step in SilkCentral Test Manager

Delete Permissions

Delete permissions for nearly all assets in SilkCentral Test Manager have been separated from the previous Manage permissions. Please review your custom roles to avoid problems.

Removing Data-Driven Properties

When you have data-driven tests in use, and you have data-driven properties assigned to specific tests in the **Tests** tree, you can now remove the properties when you no longer need them. In previous versions of SilkCentral Test Manager, you were able to deactivate the data-driven properties, but not completely remove them.

Email Hyperlink

Each topic in the online help now includes the following hyperlink at the bottom of it: *Send feedback about this topic*.

Click the link to open your Web browser and submit feedback about that page of the help.

Usability Enhancements

This section lists the usability enhancements that have been made throughout Test Manager.

Copying Filters

You can now re-use existing filters to create new filters. The new functionality is available in the toolbar of the **Requirements**, **Tests**, or **Execution Planning** area. To create a new filter out of an existing one, copy the existing filter, edit the properties of the filter, and then save the edited filter as a new filter with a different name.

Comment Execution Plan Runs

You can now add comments to execution plan runs in the **Execution Plan Runs** page. To edit an existing comment, right-click the execution plan run and select **Edit Run Comment**.

Deleting Result Files

You can now delete result files in the **Test Details** area of the **Current Run** page of a manual test.

Integration Enhancements

This section lists the enhancements that have been made to the integrations in Test Manager.

IBM Rational DOORS Version 9.3

Test Manager now supports IBM Rational DOORS Version 9.3.

Oracle 10g (version 10.2.0.5)

Test Manager now supports Oracle 10g (version 10.2.0.5).

Oracle 11g (version 11.2.0.2)

Test Manager now supports Oracle 11g (version 11.2.0.2).

VMware vCenter Lab Manager 4.0.4

Test Manager now supports VMware vCenter Lab Manager 4.0.4.

Changepoint 2010 SP1

Test Manager now supports Compuware Quality Manager with Changepoint 2010 SP1.

Bugzilla 4.0

Test Manager now supports Bugzilla 4.0.

Microsoft Office Word 2007 .docx Import

Test Manager now supports the Microsoft Office Word 2007 .docx file format.

The functionality for `.docx` is the same as for `.doc` with the following improvements:

- Images and any other embedded objects (OLE) (for example: PDF, Excel) are detected and imported as an attachment to a requirement.
- Descriptions may contain hyperlinks.

MSTest

Test Manager now supports MSTest distributed with Visual Studio 2010/Test Agent 2010.

Web Service Enhancements

This section lists the enhancements that have been made to the Test Manager Web services.

Copy Project API

You can now copy projects using the Web service API. You can specify the same settings that you can specify in the user interface.

Installing and Licensing Test Manager

For information regarding the installation and licensing of Test Manager, refer to the Test Manager Installation Help, which is also available from both the Test Manager installation CD and the Test Manager download site.

SilkCentral Issue Manager

SilkCentral Issue Manager (Issue Manager), the issue-tracking tool of SilkCentral, is fully integrated with Test Manager, enabling you to correlate issues with system requirements and executed tests.

Test issues can be added and managed in the menu through **Tests** > **Details View** > **Issues**. For more information, refer to the Issue Manager documentation.

Note: StarTeam and IBM Rational ClearQuest are also supported by Test Manager “out of the box”. Additional issue tracking systems can be configured by installing a custom plug-in. For more information, refer to the *Test Manager API Help*.

Browser Settings

To optimize the functionality and performance of Test Manager, adjust the following settings in your browser:

- Make sure that you enabled the cache of your browser.
- If you have problems accessing Test Manager through the browser, add the Test Manager server to the list of trusted sites in your browser.

For a list of supported browsers, refer to the Release Notes.

Logging In to and Out from Test Manager

The procedures in this section explain how to log in to and out of Test Manager.

Logging in to Test Manager

To log in to Test Manager:

1. Navigate to the IP address or URL of your Test Manager installation.

   Note: Speak to your system administrator about the URL, username, and password you should use to log in to Test Manager.
2. On the Test Manager login page, type in your **Username** and **Password**.
3. Check the **Remember login** check box to have Test Manager auto-complete usernames and remember your password when you begin typing your username.
4. Click **Login**.

You will be directed to your personal **Dashboard**. You can quickly **Continue working in your last visited area** by clicking this link on the top of the **Dashboard** page.

Logging out from Test Manager

To log out from Test Manager:

1. In the menu, go to **User:<Username>**.
2. Click **Log out**.

Working with SilkPerformer Projects

SilkPerformer is fully integrated with the test and execution functionality of Test Manager. SilkPerformer projects can be integrated into Test Manager tests and directly executed through Test Manager. This allows for powerful test-result analysis and reporting. It also enables unattended testing, which means tests are run automatically by Test Manager based on pre-configured schedules.

Refer to the SilkPerformer Help for details on configuring the integration of SilkPerformer with Test Manager.

SilkPerformer project files can be directly opened in SilkPerformer from Test Manager, where scripts and settings can be edited. Edited SilkPerformer projects can subsequently be checked back into Test Manager to make them available for future test executions.

Test Manager provides information on execution plan run properties during SilkPerformer test executions. Use the `AttributeGet` methods to access execution plan run properties in the SilkPerformer script. You can access the following properties in the script:

- `#sctm_execdef_id`
- `#sctm_execdef_name`
- `#sctm_product`
- `#sctm_version`
- `#sctm_build`
- `#sctm_keywords`

Note: The term Project is used differently in SilkPerformer than it is in Test Manager. A SilkPerformer project, when uploaded to Test Manager, becomes the core element of a Test Manager test. Test Manager projects are high-level entities that may include multiple SilkPerformer projects, tests, execution plans, and requirements.

Working with Silk Performance Explorer

Silk Performance Explorer (Performance Explorer) is used for in-depth analysis of test runs. Performance Explorer results analysis can be started directly from the **Execution** area and the **Tests** area of Test Manager through execution runs on the **Runs** page or from Performance Explorer itself. Refer to the Performance Explorer documentation for details regarding the integration of Performance Explorer with Test Manager.

The results of load-test runs in SilkPerformer can also be uploaded to Test Manager and associated with tests. Refer to the *SilkPerformer Help* for more details.

For additional information about the integration of Test Manager integration with SilkPerformer, refer to the *SilkPerformer Help* and the *Performance Explorer User Guide*.

SilkCentral Administration Module

SilkCentral products are based on SilkCentral Architecture (SCA), which allows for common administration of Web-based products. These administrative tasks are performed through the SilkCentral Administration Module.

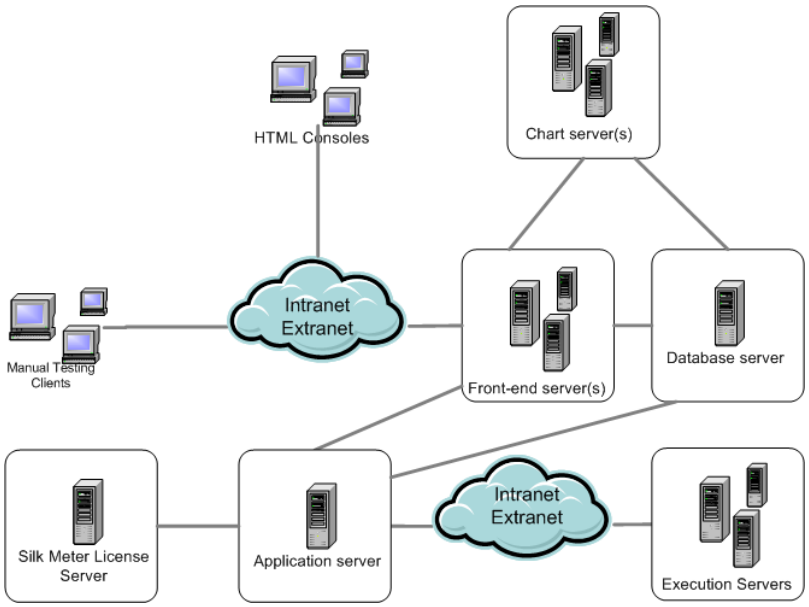
SilkCentral Architecture

SilkCentral products are based on SilkCentral Architecture (SCA), which allows for common administration of Web-based products.

The following sections describe the SilkCentral components.

- Overview
- Front-End Server
- Application Server
- Execution Server
- Chart Server
- Database Server
- SilkMeter License Server
- Manual Testing Client

Overview



Front-End Server

The front-end server is responsible for the graphical user interface. This server is based on HTML and is accessible from any Web browser, such as Internet Explorer or Firefox. A user sends an appropriate HTTP request to the front-end server and receives a login page for authentication. After successful login, the user can use the corresponding application based on the respective user rights. The front-end server can operate as a stand-alone HTTP server, or it can be attached to a Web server, such as IIS via ISAPI filter. The front-end server uses port 19120. For secure connections with SSL, the server also uses port 443.

Application Server

The application server synchronizes tasks such as the distribution of schedules, control of execution servers, and management of database configuration. These tasks require a centralized agency to ensure the consistent, reliable behavior of the application. The application server also evaluates results, saves them to the database, and sends alerts based on success conditions. The application server uses port 19122.

Execution Server

The execution server executes automated tests that are scheduled by authorized users. Users are responsible for the proper configuration of execution servers and additional resources that are required for

test executions. The system allows for the installation and configuration of multiple execution servers working independently of one another. The execution server uses port 19124. For secure connections with SSL, the server also uses port 19125.

Chart Server

The chart server is used to generate charts that are viewed in reports. The system allows for the configuration of a pool of chart servers. A built-in load balancing mechanism uses the pool to distribute chart generation. The chart server is also used to generate reports and deliver them directly to the end-user for viewing within a browser. The chart server uses port 19126.

Database Server

System persistency is implemented using a RDBMS (Relational Database Management System). The database server uses ports 1433 and 1521.

SilkMeter License Server

SilkMeter, the licensing software that accompanies Silk products, determines the SilkCentral-application functionality that you may access. For more information on licensing, refer to the installation guide of the respective product. SilkMeter uses port 5461.

Manual Testing Client

The Manual Testing Client enables testers to manage their tests and track results without the need of an Internet connection. The Manual Testing Client offers the core, manual-test execution functionality that Test Manager provides, but through a Windows-based client tool.

Agent Computers

SilkPerformer and SilkTest Classic agent computers are assigned to particular SilkPerformer or SilkTest Classic projects from the pool of agent computers that are available to the controller computer. In combination with Test Manager, the controller computer acts as an execution server.

SilkPerformer Agents	SilkPerformer agent computers host the virtual users that are run during load tests. As many agent computers as necessary can be added to a SilkPerformer project so that the required quantity of virtual users can be run. Configuration of agents is done through SilkPerformer. Refer to the SilkPerformer documentation for details on configuring agents.
SilkTest Classic Agents	The same rules that apply to SilkPerformer agents apply to SilkTest Classic agents, except SilkTest Classic agents host SilkTest Classic tests.

Access and Licensing

SilkMeter, the licensing software that accompanies Silk products, determines the SilkCentral-application functionality that you may access. For more information on licensing, see the respective product's installation instructions.

Test Manager, Risk-Based Testing, and Quality Goals

Often during software development, a balance must be struck between testing scope, time, and cost. One commonly used method in determining this balance is risk-based testing.

A key practice of risk-based testing is the process of categorizing test assets based off of factors important to the quality and to the end users of the application.

Based off of this categorization process, quality practitioners can determine the appropriate set of tests based off of time available and the desired goals of the quality process. Executing a subset of tests

involves some amount of risk. Quantifying this risk in a relevant way ensures that the stakeholders are informed throughout the process, which is key for a successful implementation of risk-based testing.

In Test Manager, factors used for risk-based testing are completely customizable using *Quality Goals*. A quality goal can be created for any **List** type custom attribute or requirement property which are used to categorize testing assets. This allows users to define a subset of tests that need to be executed in order for the project to meet specific quality criteria. Quality goal reporting allows users to communicate to stakeholders the test planning and execution process, relevant scope, time, and quality metrics on testing assets.

Implement a Risk-Based Testing Approach by using Quality Goals

For a given project, a user wants to balance the scope of testing and time while still ensuring the application is of acceptable quality. The following steps illustrate a high-level workflow of how a user can use quality goals to implement a risk-based testing approach.

In this project, the stakeholders are interested in ensuring enough of the test bed has been executed as well as ensuring the most important requirements have been sufficiently tested. The data for the below example is also included in the `Demo` project that is included with new versions of Test Manager.

1. Select a project.
2. Create a requirement property of type **List** to capture the importance of the requirement.

   For this example, create the requirement property `Business Value` with values:

   - `Low`
   - `Medium`
   - `High`
3. Create a test attribute of type **List** to capture the type of purpose of the tests.

   For this example, create the test attribute `Level` with values:

   - `Full`
   - `Regression`
   - `Smoke`
4. Click **Reports** > **Quality Goals**.
5. Create a new quality goal.

   For this example, create the quality goal `Business Value` for requirement property with the following **Goal %**:

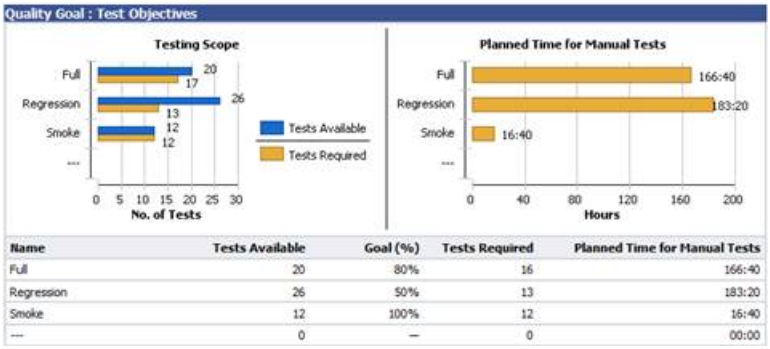
   - `High-100%`
   - `Medium-75%`
   - `Low-50%`
6. Create a new quality goal.

   For this example, create the quality goal `Test Objectives` for the test attribute with the following **Goal %**:

   - `Full-80%`
   - `Regression-50%`
   - `Smoke-100%`
7. After the test assets are developed, assign the attributes/properties to the requirements and tests.
8. To determine the testing scope and ensure stakeholder agreement in the testing scope, click **Reports** > **Details View**.
9. Click **Tests** > **Status Report** > **Quality Goals Planning Report**.

   This report will show you the number and percentage of tests for each quality goal value, total planned time for manual tests, and the number of tests required to meet the goal vs. the number of test available in the project.

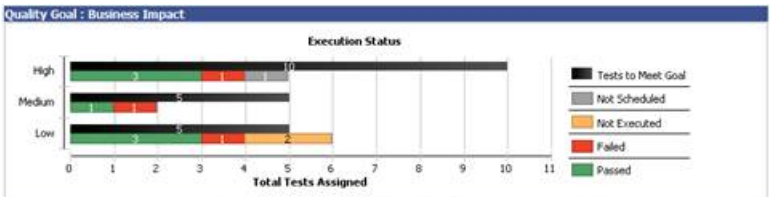
Name	Tests Available	Goal (%)	Tests Required	Planned Time for Manual Tests
High	10	100%	10	83:20
Medium	6	75%	5	00:00
Low	9	50%	5	66:40
---	0	–	0	00:00



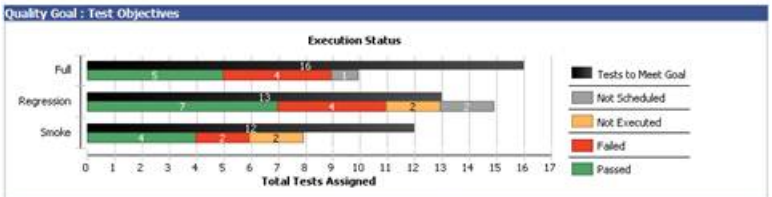
Name	Tests Available	Goal (%)	Tests Required	Planned Time for Manual Tests
Full	20	80%	16	166:40
Regression	26	50%	13	183:20
Smoke	12	100%	12	16:40
---	0	–	0	00:00

10. After agreement on the plan has occurred and the project is progressing through testing execution, testing status for the Quality Goals can be easily evaluated using the **Quality Goals Execution Report**.

11. Click **Tests** > **Status Report** > **Quality Goals Execution Report**.



Name (Goal %)	Total Tests Assigned: Passed	Failed	Not Executed	Not Scheduled	Total	Tests to Meet Goal	Execution % to Goal
High (100%)	3	1	0	1	5	10	40%
Medium (75%)	1	1	0	0	2	5	40%
Low (50%)	3	1	2	0	6	5	80%



Name (Goal %)	Total Tests Assigned: Passed	Failed	Not Executed	Not Scheduled	Total	Tests to Meet Goal	Execution % to Goal
Full (80%)	5	4	0	1	10	16	56%
Regression (50%)	7	4	2	2	15	13	85%
Smoke (100%)	4	2	2	0	8	12	50%

Quick Start Tasks

This section includes the quick start tasks that are available in Test Manager. Quick start tasks are high-level overviews of the main tasks that you will likely need to perform with Test Manager. These tasks can serve as tutorials in guiding you step-by-step through the best practice usage of the core functionality of Test Manager.

Managing Shared Step Libraries - Quick Start Task

To manage your shared step libraries, you will need to perform some or all of the following tasks.

Creating a Shared Step Library

1. In the menu, click **Tests** > **Libraries of Shared Steps**.
2. In the **Libraries** tree, select the root node, which is called **Shared Steps Libraries**.
3. In the toolbar, click .

   Note: You can also right click on the root node and click **New Library**.

Creating a Shared Steps Object

You can create a shared steps object in the **Steps** page of any manual test or shared steps object.

To create a shared steps object:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Navigate to the **Steps** page of the manual test or shared steps object that includes the steps you want to share.
3. Select the steps with `CTRL + CLICK` or `SHIFT + CLICK`.
4. Click . The **Create Shared Steps** dialog box opens.
5. Enter a name for the new shared steps object in the **Name** text box.
6. Choose the location where you want to place the new shared steps object in the **Libraries** tree.
7. Click **OK**. The shared steps object is created and the selected steps are replaced by a call to the shared steps object.

Adding a Call to Shared Steps

To add a call to a shared steps object:

1. In the menu, click **Tests** > **Details View** to add the shared steps to a test or click **Tests** > **Libraries of Shared Steps** to add the shared steps to another shared steps object.
2. In the corresponding tree, click on the node to which you want to add the shared steps.
3. Click the **Steps** tab.
4. To add the shared steps to the end of the steps list, click . To insert it above the selected step, click . The **Call to Shared Steps** dialog box opens.
5. Select the shared steps node, whose steps you want to add, from the tree.

   Note: For a test, the tree displays only the libraries that are visible to the active project. For more information, see the *Visibility Page*.
6. Click **Ok**.

Editing Manual Test Steps

Edit the manual test steps of a test or a shared steps object in the **Steps** page.

To edit a manual test step:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the node whose steps you want to edit in the **Tests** or **Libraries** tree.

3. Click the **Steps** tab.
4. Click on the test step that you want to edit. The details of the test step are shown in the **Edit Step** view.
5. Edit the name, action description, expected results, and step properties of the test step in the corresponding text boxes.

   Note: You can insert values from data sources into manual test steps in the form of parameters.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
6. Perform one of the following actions to finish editing the test step:
   - Click **Ok and New Step** to save your changes and add a new step after the current.
   - Click **Ok** to save your changes.
   - Click **Reset** to revert your changes.

Creating a Version

To be able to revert a set of changes you want to apply to a shared steps object or a manual test, create a version of the element.

To create a version of a shared steps object or a manual test:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the shared steps object in the **Libraries** tree, or the manual test in the **Tests** tree.
3. Click in the toolbar.

   This action is only enabled if the shared steps object or the manual test was changed.

   The **Create Versions** dialog box opens.
4. *Optional:* Enter a comment on the version in the **Comment** text box.
5. Click **Ok**. The new version is shown in the **History** page.

Comparing Versions

To view the differences between two versions of a shared steps object or a manual test, compare the versions.

To compare two versions of a shared steps object or a manual test:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the shared steps object in the **Libraries** tree, or the manual test in the **Tests** tree.
3. Click the **History** tab.
4. Select the two versions you want to compare with **CTRL + CLICK**.
5. Right click on the selection.
6. Select **Compare Versions**. The **Compare Versions** dialog box opens, displaying all the added, changed, and removed content.

   Note: Use the arrows to change the version numbers and compare additional versions.

Reverting to a Previous Version

To undo changes to a shared steps object or a manual test, revert to a previous version.

To revert a shared steps object or a manual test to a previous version:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the shared steps object in the **Libraries** tree, or the manual test in the **Tests** tree.
3. Click the **History** tab.
4. Right-click on the version to which you want to revert the shared steps object to.
5. Click **Revert to Version <ID>**.
6. The shared steps object or manual test is reverted to the selected version and a new version is created in the **History** page. Attachments, parameters, and data sets are not reverted for manual tests.

Configuring Projects - Quick Start Task

To configure a project, you will need to perform some or all of the following tasks.

Configuring Project Settings

To customize the project settings:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Project Settings** tab to view the current settings. The **Project Settings** page opens.
3. Click **Edit** to modify the current project settings.
4. The **Edit Project Settings** dialog box displays. You can specify the following information:

Build Information File Name	Build information files contain project information, including build number, build log location, error log location, and build location. Enter the name of your project's build information file in this text box. All test executions will read the build information from this specified file.
Project Release Date	Specify the planned release date for your project.
File Extensions to ignore in Results	Specify result file types or other file types that should not be saved as results for test executions.

   Note: File extensions must be separated by commas, for example, xlg, *_, res. Changes made in the **Build Information File Name** and **File Extensions to ignore in Results** fields will not effect scheduled tests. To redistribute tasks to execution servers, you must reschedule the tests, or disconnect from and reconnect to the database.
5. Click **Save** to save your project settings.

Creating Custom Attributes

To create a custom attribute:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Attributes** tab to view the list of current attributes.
3. Click **New Attribute**. The **New Attribute** dialog box displays.
4. Type a **Name** for the new attribute. This name will be displayed in list boxes when the attribute becomes available for use.
5. Type a **Description** for the new attribute.
6. Select the attribute **Type**. See the *Test Attribute and Requirement Property Types* topic for descriptions of each type of attribute.
7. Click **OK**.

Creating Filters

To create a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Filters** tab to view the list of available filters.
3. Click **New Filter**. The **New Filter** dialog box displays.
4. Type a **Name** for the new filter.

   This name will be displayed in list boxes when the filter becomes available.
5. Select a **Category** for the new filter from the list box to make the filter available in a specific Test Manager area.

Category	Area
Requirement	Requirements
Test	Tests
Execution	Executions

6. Type a **Description** for the new filter.
7. *Optional:* Check the **Visible to other users** check box to allow other users to see the filter.
8. *Optional:* Check the **Editable by other users** check box to allow other users to edit the filter.

   Note: If **Visible to other users** and **Editable by other users** are checked, the filter is public. To delete private filters, you have to be the owner of the filter or you need the **Delete private filters of other users** permission.
9. Select a category of filter criteria from the **Selection criteria** list box. The available categories depend on the general filter category you have selected.

   Note: You can combine filters by selecting **Nested Test Filter** or **Nested Requirements Filter**. Selecting one of these categories allows you to include an existing filter in your new filter.
10. Select a **Property**, **Operator**, and **Value** for the filter from the respective list boxes.

    Property Available properties depend on the filter category that you have selected in the previous step. Defines the property for which you are defining a filter setting. If you have selected an attribute category, the property list includes custom attributes to query against.

    Operator Specifies the filter operator. The operator depends on the property type you have selected. For example, if you have selected a property that is based on a string field type, the following operators are available:

Operator	Description
=	The string equals the defined value.
not	The string is different than the defined value.
contains	The string contains the defined value.
not contains	The string does not contain the defined value.

    Value Enter the value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.
11. *Optional:* Click **More** if you want to add more than one filter category to the new filter. Repeat this procedure to define new categories.

Note: If you define more than one filter category, you must define whether the categories need to be fulfilled in addition to the existing categories (AND relationship), or if the filter returns true when one of the filter categories is fulfilled (OR relationship). Select either **AND** or **OR** to define the filter category relationship. You cannot define nested AND, OR relationships.

12. *Optional:* To remove filter categories, click **Fewer**. This removes the last filter category.
13. Click **OK** to save the new filter, or click **Cancel** to abort the operation.

Enabling Change Notification

To enable change notification:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Notifications** tab.
3. Click **Configure Notification** to open the **Configure Change Notification** dialog box.
4. If you want to be notified by email when changes are made to requirements in the currently selected project, check the **Changes on Requirements** check box.
5. If you want to be notified by email when changes are made to tests within the currently selected project, check the **Changes on Tests** check box.
6. Click **OK** to save the notification settings, or click **Cancel** to abort the operation without saving changes.

You will be notified by email about changes for which you have activated notification.

Creating Step Properties

To create a new step property:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Step Properties** tab.
3. Click **New Property**. The **New Step Property** dialog box opens.
4. Enter a name for the new property in the **Name** text box.

   Note: Step property fields are always declared as type string.
5. Click **OK** to make your property available to all manual test steps in the selected Test Manager project.

   Note: To create a step property for a library, select the library node in the **Libraries** tree and continue with the second step.

Analyzing Test Results - Quick Start Task

To analyze the results of a test, you will need to perform some or all of the following tasks.

Creating New Reports

To create a new report:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the folder in which you want the new report to display.
   This determines where the report is stored in the directory tree.
3. Click on the toolbar. The **Create New Report** dialog box opens.
4. Type the name of the new report.

This is the name that is displayed in the **Reports** tree.

5. Check the **Share this report with other users** check box if you want to make this report available to other users.
6. In the **Timeout [s]** field, type the maximum time period in seconds that Test Manager should wait for SQL queries to complete.
7. From the **Default tab** list box, select the tab that you want to be directed to when you select this report from one of the context-sensitive report lists.
8. Select the corresponding result type from the **Result category** list box.

   This setting specifies the database table and view that is to be filtered for the report. The following result types are available:

Result Type	Description
Requirement	Returns requirements available in the **Requirements** module that meet the query criteria.
Test	Returns tests available in the **Tests** area that meet the query criteria.
Test Execution	Returns executed test results from the **Executions** area that meet the query criteria.
Execution Plan	Returns execution plans from the execution area.
Issue	Returns issues, including imported issues.
Requirement Progress Builds	Contains information on requirements progress per build so that you can see how requirements develop across builds.
Requirement Progress Days	The same as Requirement Progress Builds, but shows development on a daily basis.
Test Progress Builds	Shows how tests develop across builds.
Test Progress Days	Same as above, but shows development on a daily basis.

   Each result type offers a set of selection criteria. Based on the result type you have selected, specify an appropriate **Selection criteria** for your report. These criteria typically group properties based on a view or some other intuitive grouping, for example custom properties.
9. From the **Property** list box, select the property that is to be filtered on.

   For some selection criteria, properties are dynamic.
10. Select an **Operator** for the query.

    The available operators depend on the property. Example operators are =, `not`, `like`, and `not like`. Strings are always compared lowercase. Allowed wildcards for strings are "*" and "?", where * matches any characters and ? matches exactly one character.
11. Select or specify the **Value** that the query is to be filtered on.

    For date-based properties, the **Value** field is replaced with a calendar tool that you can use to select a specific date.
12. *Optional:* To add an additional query string to this report, click **More**. An existing query string can be deleted by clicking the string's **Delete** button. When multiple query strings are defined, **AND** and **OR** option buttons are displayed next to **More**. Use these option buttons to define if the queries should be considered cumulatively, or if only one query string's criteria needs to be met.
13. Click **Next** to configure report columns on the **New Report** dialog box.
14. Click **Add Columns**. The **Add Columns** dialog box lists all available report columns.
15. Select the columns that you want to have included in the report and click **OK**.

    You can select multiple columns with **CTRL + CLICK**.

Note: For test-planning reports, the list of available column names is enhanced with the column names from the *LQM_v_tests* table. For details, refer to the *Test Manager Database Model Help*.

The selected columns display in tabular format on the **New Report** dialog box.

16. *Optional:* Configure how each report column is to be displayed. For each column, specify a sort direction, `ascending`, `descending`, or `unsorted`, using the up/down arrows in the **Sorting** column.
17. When multiple columns are selected for sorting, a list box is displayed in the **Sort Order** column that allows you to more easily edit the column-sort order. Set these numbers as required.
18. Give each column an **Alias**.

    This is the name by which each column will be labeled in the generated report.
19. With grouping, you can take advantage of SQL aggregation features, for example when selecting a number of elements or querying a total sum of values. Check the **Group by** check box to specify that SQL group by functions are to be applied.
20. Columns that are not selected for SQL group by functions are set to aggregation by default, which means a single aggregate value is calculated. From the **Aggregation** list box, select the appropriate aggregation type.

    The following types are available:

    - Count
    - Sum
    - Average
    - Minimum
    - Maximum
21. The **Actions** column enables you to move column listings up and down in the view, or to delete a column. The **Move Up** and **Move Down** functions do not affect the outcome of the report.
22. Click **Finish** to complete your new report.

Editing Report Properties

To edit the properties of a report:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Report** tab.
4. Click the **Properties** tab.
5. Click **Edit**. The **Edit Report** dialog box displays.
6. Modify the **Name** and **Description** of the report as required.
7. Ensure that the **Share this report with other users** check box is checked if you intend to have this report shared with other users.
8. From the **Default tab** list box, select the tab that you want to be directed to when you select this report from one of the context-sensitive report lists.
9. Specify one of the following options to indicate how the report can be edited:

Option	Description
Simple report	You can modify the selection criteria—thus changing the results of the selected report—or you can click **Advanced Query** to modify the SQL query code.
Advanced report	If you have familiarity with SQL, you may edit the query code in the **Report data query** text box. To assist you in editing SQL queries, a list box of function placeholders (for example, “variables”) is available. To insert one of the available pre-defined functions, select the corresponding placeholder from the **Insert placeholder** list box.

- *Advanced report:* If you have familiarity with SQL, you may edit the query code in the **Report data query** field. To assist you in editing SQL queries, a list box of function placeholders, for example

"variables", is available. To insert one of the available pre-defined functions, select the corresponding placeholder from the **Insert placeholder** list box.

Note: If you manually edit the SQL code for the query, upon finishing, click **Check SQL** to confirm your work.

10. Click **Finish** to save your changes.

Editing Report Parameters

To edit the parameters of a report:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Parameters** tab. If the report has parameters defined for it, the parameters are listed here.
4. Click **Edit Parameters**. The **Edit Parameters** dialog box displays.
5. Edit the **Label** or **Value** of the listed parameters as required.
6. From the **Usage** list box, select the usage type of the parameter.
   - constant value
   - start time
   - end time
7. Click **OK** to save your changes.

Writing Advanced Queries with SQL

Advanced reports can be created through manual SQL coding. Virtually any reporting option is available if you know the database schema. Clicking **Advanced Query** hides the query string list boxes and opens the **Report data query** text box in which you can insert existing code or write new SQL code.

One approach is to begin query-string construction using the list boxes as outlined in *Creating New Reports*. If the report criteria are valid, the equivalent SQL statement will be generated and displayed, and then move to advanced mode for further modifications.

Note: You cannot move from advanced mode back to simple mode.

To write an advanced query directly in SQL:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the folder in which you want the new report to display.
   This determines where the report is stored in the directory tree.
3. Click on the toolbar. The **Create New Report** dialog box opens.
4. Type the name of the new report.
   This is the name that is displayed in the **Reports** tree.
5. Check the **Share this report with other users** check box if you want to make this report available to other users.
6. Type a description of the report in the **Description** text box.
7. Click **Advanced Query** to open the **Report data query** text box. Insert previously written code as necessary, or write new code directly in the text box.
   To assist you in writing SQL queries, a list box of Test Manager function placeholders is available. For details, see *SQL Functions for Custom Reports*. To insert one of the available pre-defined functions, select the corresponding placeholder from the **Insert placeholder** list box.

   Note: If you manually edit SQL code for the query, click **Check SQL** to confirm your work.
8. Once you have completed editing the report's properties, click **Finish** to save your settings.

Customizing BIRT Report Templates

With BIRT RCP Designer (BIRT), you can customize the pre-installed report templates of Test Manager and create custom report templates. For details on using BIRT, refer to the *SilkCentral Administration Module Help* and the BIRT RCP Designer documentation.

To download an existing template for editing:

1. In the menu, click **Reports** > **Details View**.
2. Select a report that utilizes the BIRT Report Template.
3. Click the **Properties** tab.
4. Click **Download BIRT report template**. You receive the report data as an empty generic BIRT report template. The datasource is already configured.
5. Once you have saved the template to your local system, modify it as required.

   For detailed information on configuring BIRT report templates, refer to the *SilkCentral Administration Module Help*.
6. Upload the modified report template by using the **Upload** link on **Administration** > **Reports** > **Report Templates**.

Adding Sub-Reports

To aggregate the results from multiple reports into the currently selected report, you can add sub-reports. When adding a report as a sub-report, the result columns and rows of the sub-report are concatenated to the results of the selected report.

To add a report as a sub-report:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Properties** tab.
4. Click **Add Sub-Report**.

   The **Add Sub-Report** dialog box displays.
5. From the **Reports** tree, select the sub-report you want to append to the current report.
6. Click **OK** to complete the addition of the sub-report. Sub-reports are displayed on the associated report's **Properties** page in the **Sub-Reports** section.

Viewing Reports

Because each template expects a certain data format to produce a useful graph, not all templates can be applied to all report queries. You will receive an error message if you attempt to generate a report through an incompatible report template. For example, selecting the *Four Values Per Row As Horizontal Bar* template to display the *Requirements Status Overview* report works because this particular Microsoft Excel template requires exactly the four values, `failed`, `passed`, `not executed`, and `not covered` that the report query delivers.

To generate a report:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the report that you want to generate.
3. Click the **Report** tab.
4. Click the **Select Report Template** icon. The **Select Report Template** dialog box opens.
5. Select the template you wish to use.
6. Click **OK** to display the report.
7. *Optional:* If necessary, select an alternate view magnification for the report from the list box.

   100% is the default magnification. Other options are 50%, 75%, 150%, and 200%.

Displaying Charts

To display a chart:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree for which you want to view a chart.
3. Click the **Chart** tab to display the default chart.
4. To select a required chart type, click .
5. On the **Select Chart Type** dialog box, select a chart type from the list box.
6. Check the view properties that you want to apply to the chart.
   - 3D view
   - Show horizontal grid lines
   - Show vertical grid lines
   - Show legend
7. Specify how these chart options are to be saved:
   - Click the **For current user only** option button to have these chart settings override the report's standard settings whenever the current user views this chart.
   - Click the **As report standard** option button to have these chart settings presented to all users who don't have overriding user settings defined. This setting does not effect individual user settings.
8. Click **OK** to display the new chart type.

Note: The chart configurations you define here become the defaults for this report. When standard charts and graphs are not able to deliver the specific data that you require, or when they cannot display data in a required format, you can customize the appearance of queried data using the Test Manager reporting functionality. To open the current chart in a separate browser window, click at the top of the **Chart** page.

Generating Code-Change Impact Reports

To generate a code-change impact report:

1. In the menu, click **Projects** > **Project List**.
2. Select the project for which you want to analyze code-coverage data.
3. In the menu, click **Reports** > **Code Analysis**.
4. Click **Create Code Change Impact Report** on the toolbar. The **Select Classes for Report** dialog box displays.
5. Select a **Product** and **Version**, if you want to change the pre-selected values.

   Click **Add** to add the class file(s) to the **Selected classes** pick list.

   You can remove classes in the **Selected classes** pick list by selecting entries and clicking **Remove**. Click **Remove All** to remove all selected classes from the **Selected classes** pick list.
6. In the **Filter** text box, type criteria to filter the packages/namespaces. For example, entering the string "published" will only list packages/namespaces that contain the string "published" in their names.
7. Select a package from the **Packages/Namespaces** list box.

   Use **CTRL + CLICK** to select multiple packages/Namespaces.

   The classes that are available in the selected package/namespace are displayed in the **Classes** list box.
8. In the **Classes** list box, select a class that you want to have included as a source in your report.

   Use **CTRL + CLICK** to select multiple classes.
9. Click **Add**. The selected classes are added to the **Selected classes** list box.
10. Repeat the preceding steps to add the class files to the **Selected classes** list box until you have added all required classes to the **Selected classes** list.

11. Select a report from the **Select report** list box.
12. Click **OK** to generate the report.

Managing Requirements - Quick Start Task

To manage your requirements with Test Manager, you will need to perform some or all of the following tasks.

Creating Requirements

Note: Newly created Test Manager projects do not contain requirements.

To create a new requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Click **New Requirement** on the toolbar.

   Note: If the project you are working with does not yet have any requirements associated with it, click the **<Click here to add Requirements>** link in the **Requirements** tree to open the **New Requirement** dialog box.

   The **New Requirement** dialog box opens.
3. Type a meaningful **Name** and **Description** for the requirement.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for description text boxes.
4. Select the appropriate **Priority**, **Risk**, and **Reviewed** status from the list boxes.
5. If custom requirements have been defined, they are listed below the **Reviewed** list box. Type in the corresponding text box any custom property data that you want tracked with this requirement.

   Note: The **Priority**, **Risk**, **Reviewed**, and any **Custom Property** fields will be configured automatically with the corresponding properties of the parent requirement if you check the **Inherit from parent** check boxes for these properties.
6. Click **OK** to create a new top-level requirement. Alternatively, you can click **OK and New Requirement** to both save the newly created requirement and open the **New Requirement** dialog box to create an additional top-level requirement. Or, you can click **OK and New Child Requirement** to have the **New Child Requirement** dialog box open after the new top-level requirement is created.

Configuring Requirement Types

A requirement without a configured requirement type is not available for upload to a requirements management system (RMS). When you import a requirement from an RMS, the appropriate requirement type is automatically configured.

Note: For requirements with the type CaliberRM, Requisite Pro, and DOORS, you can only configure the requirement type for top-level tree elements, which are requirements that are a direct child of the project node. All other requirements share the requirement type of their parents.

To configure the type of a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Click in the toolbar.
3. Click the **Properties** tab. The **Properties** page displays the properties of the selected tree-element.
4. Click **Map Requirement** to select a requirement type from the list. Requirement type is a categorization used by CaliberRM, Requisite Pro, and DOORS, and is required for synchronization.

   Note: Map Requirement is only enabled when external requirements integration is enabled in the **Requirements Management** page in **Project:<Project Name>** > **Project Settings** and the

requirement has not been uploaded to the external requirements management tool. Additionally, the option **Enable upload of requirements to...** must be enabled.

5. Click **OK** to save your settings and close the dialog box.

Attaching a File to a Requirement

To attach a file to a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click the **Attachments** tab.

   When requirements management integration has been enabled between a Test Manager project and a CaliberRM project, the **Attachments** page includes an **Open CaliberRM** button, which enables you to manage requirement attachments directly in CaliberRM.
4. Click **Upload File** The **Upload File** dialog box opens.
5. Click **Browse** and select the file that you want to attach from your local file system.
6. Type a **Description** for the attachment.
7. Click **OK**. The attachment is uploaded to the server and associated with the selected requirement.

Note: Attaching files to a requirement may not work in Mozilla Firefox. Firefox requires usage of three slashes, for example `file:///`, for a file link, while other browsers require only two, for example `file://`. Aditionally, Firefox includes a security feature blocking links from remote files to local files and directories. For more information, see *http://kb.mozillazine.org/Firefox_:_Issues_:_Links_to_Local_Pages_Don't_Work*

Attaching a Link to a Requirement

To attach a link to a requirement

1. In the menu, click **Requirements** > **Details View** .
2. Click in the toolbar.
3. Select the requirement in the **Requirements** tree.
4. Click the **Attachments** tab.

   When requirements management integration has been enabled between a Test Manager project and a CaliberRM project, the **Attachments** page includes an **Open CaliberRM** button, which enables you to manage requirement attachments directly in CaliberRM.
5. Click **Attach Link**. The **Attach Link** dialog box opens.
6. Type the URL in the **Link** text box.
7. Type a **Description** for the attached link.
8. Click **OK**. The link is associated with the selected requirement.

Generating Tests from Requirements Details View

You can generate tests directly out of the **Requirements** tree and assign tests to specific requirements. The **Requirements** tree serves as a template for the test-folder/test structure of the new **Tests** tree.

To generate a new test from **Details View**:

1. In the menu, click **Requirements** > **Details View** .
2. Right-click the requirement or project node that you want to convert into a test and select **Generate Test**. The **Generate Test from Requirements** dialog box displays. This dialog box enables you to specify whether the leaves, which means the lowest-level nodes, of the selected requirements sub-tree should be converted into tests or test folders and whether the tree should be generated into a new test container or an existing container.

3. Enter a name for the new test container in the **Enter Name** text box and select a product from the **Select Product** list box to create the container within the active Test Manager project.

   The **Select Product** list box is populated with the products that are configured by a project manager. For detailed information, refer to the *SilkCentral Administration Module Help* or ask your project manager.
4. If you have defined a source control profile, select the source control profile you want to use for managing the test sources from the **Select Source Control Profile** list box.

   For detailed information on source control profiles, see *Source Control Profiles* or ask your Test Manager administrator.
5. To include all child requirements of the selected requirement in the test, check the **Include child requirements** check box.

   The check box is checked by default.
6. To have the new tests automatically assigned to the requirements from which they are created, check the **Assign newly generated Tests to Requirements** check box.

   If this option is not selected, tests must be manually associated with requirements.

   Note: This option is not available when checking **Generate Test Folders from Requirement Tree leaves**.
7. Click **OK** to create the test.

   The new test has the same structure as the **Requirements** tree.
8. A message box displays, asking if you want to switch directly to the **Tests** area. Click **Yes** to view the test in the **Tests** area, or click **No** to remain in the **Requirements** area.

Creating Filters

To create a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Filters** tab to view the list of available filters.
3. Click **New Filter**. The **New Filter** dialog box displays.
4. Type a **Name** for the new filter.

   This name will be displayed in list boxes when the filter becomes available.
5. Select a **Category** for the new filter from the list box to make the filter available in a specific Test Manager area.

Category	Area
Requirement	Requirements
Test	Tests
Execution	Executions

6. Type a **Description** for the new filter.
7. *Optional:* Check the **Visible to other users** check box to allow other users to see the filter.
8. *Optional:* Check the **Editable by other users** check box to allow other users to edit the filter.

   Note: If **Visible to other users** and **Editable by other users** are checked, the filter is public. To delete private filters, you have to be the owner of the filter or you need the **Delete private filters of other users** permission.
9. Select a category of filter criteria from the **Selection criteria** list box. The available categories depend on the general filter category you have selected.

Note: You can combine filters by selecting **Nested Test Filter** or **Nested Requirements Filter**. Selecting one of these categories allows you to include an existing filter in your new filter.

10. Select a **Property**, **Operator**, and **Value** for the filter from the respective list boxes.

Property Available properties depend on the filter category that you have selected in the previous step. Defines the property for which you are defining a filter setting. If you have selected an attribute category, the property list includes custom attributes to query against.

Operator Specifies the filter operator. The operator depends on the property type you have selected. For example, if you have selected a property that is based on a string field type, the following operators are available:

Operator	Description
=	The string equals the defined value.
not	The string is different than the defined value.
contains	The string contains the defined value.
not contains	The string does not contain the defined value.

Value Enter the value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.

11. *Optional:* Click **More** if you want to add more than one filter category to the new filter. Repeat this procedure to define new categories.

Note: If you define more than one filter category, you must define whether the categories need to be fulfilled in addition to the existing categories (AND relationship), or if the filter returns true when one of the filter categories is fulfilled (OR relationship). Select either **AND** or **OR** to define the filter category relationship. You cannot define nested AND, OR relationships.

12. *Optional:* To remove filter categories, click **Fewer**. This removes the last filter category.
13. Click **OK** to save the new filter, or click **Cancel** to abort the operation.

Creating Advanced Filters

Advanced custom filters enable you to combine simple filters to create complex filters that apply multiple filter criteria simultaneously.

To create an advanced custom filter:

1. In the menu, select the appropriate area, **Execution Planning**, **Requirements**, or **Tests**.
2. Create a new custom filter.

   For additional information, see *Creating Filters*.
3. Click **More** to display a second set of filter-parameter fields with which you can define a second set of filter parameters.
4. Select a logical operator for the application of the filtering queries. For example, with the operator `and` filtered elements must meet both sets of criteria and with the operator `or` filtered elements must meet one, but not both, of the criteria sets.
5. To delete a filter-parameter string, click ✕.
6. To display additional filter-parameter fields and create additional filter queries, click **More**. To remove excess filter-parameter sets, click **Fewer**.

Generating Tests from Requirements Details View

You can generate tests directly out of the **Requirements** tree and assign tests to specific requirements. The **Requirements** tree serves as a template for the test-folder/test structure of the new **Tests** tree.

To generate a new test from **Details View**:

1. In the menu, click **Requirements** > **Details View** .
2. Right-click the requirement or project node that you want to convert into a test and select **Generate Test**. The **Generate Test from Requirements** dialog box displays. This dialog box enables you to specify whether the leaves, which means the lowest-level nodes, of the selected requirements sub-tree should be converted into tests or test folders and whether the tree should be generated into a new test container or an existing container.
3. Enter a name for the new test container in the **Enter Name** text box and select a product from the **Select Product** list box to create the container within the active Test Manager project.

   The **Select Product** list box is populated with the products that are configured by a project manager. For detailed information, refer to the *SilkCentral Administration Module Help* or ask your project manager.
4. If you have defined a source control profile, select the source control profile you want to use for managing the test sources from the **Select Source Control Profile** list box.

   For detailed information on source control profiles, see *Source Control Profiles* or ask your Test Manager administrator.
5. To include all child requirements of the selected requirement in the test, check the **Include child requirements** check box.

   The check box is checked by default.
6. To have the new tests automatically assigned to the requirements from which they are created, check the **Assign newly generated Tests to Requirements** check box.

   If this option is not selected, tests must be manually associated with requirements.

   Note: This option is not available when checking **Generate Test Folders from Requirement Tree leaves**.
7. Click **OK** to create the test.

   The new test has the same structure as the **Requirements** tree.
8. A message box displays, asking if you want to switch directly to the **Tests** area. Click **Yes** to view the test in the **Tests** area, or click **No** to remain in the **Requirements** area.

Managing Test Executions - Quick Start Task

To manage the execution of a test, you will need to perform some or all of the following tasks.

Creating Execution Plans

To create an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select an existing folder or testing cycle in the **Execution Plans** tree, or select the project node.
3. Click on the toolbar or right-click the selected node and choose **New Child Execution Plan**. The **New Execution Plan** dialog box displays.
4. Type a name and description for the execution plan.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
5. Select a test container from the **Test Container** list box.

   You can only associate one test container to an execution plan, and you can only execute tests from this single test container.

   The latest version and build that are defined within the product that the container belongs to are populated automatically in the **Version** and **Build** list boxes.
6. Select a product **Version** and **Build** from the list boxes.

These are used when a new run of this execution plan is started. Alternatively, you can check the **Read from Build Information file** check box, in which case the version and build are read, at the beginning of each run, from the build information file on the application server. If a build information file is available on the execution server, this file is used by default for the test run, overriding the settings on the **New Execution Plan** dialog box.

7. Select a **Priority** for the execution plan from the list box.

   The priority parameter specifies the lowest priority that is considered in the data.

8. *Optional:* In the **Source Control Label** text box you can specify that earlier versions of automation files, instead of the latest versions, are fetched from the source control system.

   Note: The **Source Control Label** property is only enabled if the associated test container uses a source control profile that supports versioning.

9. Click **OK** to update the **Execution** tree with the newly created execution plan.

   Note: When you create data-driven tests, and a test has each data row as a single test, the execution plan includes a separate test for each data row. To create an execution plan with only a selection of data-driven tests, you need to assign tests with the filter option.

Manually Assigning Tests to Execution Plans

The tests that are assigned to the selected execution plans are listed on the **Assigned Tests** page.

To manually assign tests to an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan to which you want to assign the selected tests.
3. Click the **Assigned Tests** tab.
4. Click the **Manual assignment** option button. All tests of the test container which is associated with the selected execution are displayed in the **Tests** tree. If you have created a test filter, you can select it from the filter list above the **Tests** tree. To create a new test filter, navigate to the **Tests** area and click **New Filter** on the toolbar.
5. In the tree, click ⇦ left of the test that you want to assign to the currently selected execution plan.

   Clicking the assign arrow of a folder or the top-level container assigns all child tests of that parent to the selected execution plan.

Assigning Tests from Grid View to Execution Plans

The tests that are assigned to the selected execution plans are listed on the **Assigned Tests** page.

To assign one or more tests from **Grid View** to one or more execution plans:

1. In the menu, click **Tests** > **Grid View** .
2. Select the tests you want to assign to execution plans.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple tests using standard browser multi-select functions.

3. Right-click the selected tests and choose **Save Selection**.
4. In the menu, click **Execution Planning** > **Details View**.
5. Select the execution plan to which you want to assign the selected tests.
6. Click the **Assigned Tests** tab.
7. Click **Assign Saved Selection**.

   Note: Only tests that reside in the test container of the execution plan are inserted. You can insert the selected tests to more than one execution plans. You can not insert them into requirements in a different project. The selection persists until you make a different selection or close Test Manager.

Assigning Tests to Execution Plans through a Filter

The tests that are assigned to the selected execution plans are listed on the **Assigned Tests** page.

To use a filter to assign one or more tests to an execution plan:

1. Create a filter in the **Tests** area.
   See *Creating Filters* for details. If the filter already exists, skip this step.
2. In the menu, click **Execution Planning** > **Details View**.
3. Select the execution plan to which you want to assign tests.
4. Click the **Assigned Tests** tab.
5. Select **Assignment by filter** from the test assignment types.
6. Choose the filter from the list box.

Note: If you assign tests to an execution plan in **Tests** > **Grid View** , the test assignment type is automatically set to **Manual Assignment**, but the previously-filtered tests remain in the **Assigned Tests** page.

Creating a Custom Schedule

To create a custom schedule for a selected execution plan, folder, or configuration suite:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite for which you want to configure a custom schedule.

   Note: To save an edited version of a global schedule as a custom schedule, click **Edit** while the global schedule is selected in the list box. This enables you to edit the global schedule and save the result as a custom schedule.
3. Click the **Schedule** tab.
4. Click the **Custom** option button to enable the scheduling controls.
5. Click **Edit**.
6. Click next to the **From** field and use the calendar tool to specify the time and date when the execution schedule should begin.
7. Specify the **Interval** at which the tests should be executed.
8. In the **Run** section, specify when the schedule schould end.
   Select on of the following options:
   - Click **Forever** to define a schedule with no end.
   - Click **n Time(s)**.
   - Click next to the **until** field and use the calendar tool to specify the time and date when the execution schedule should end.
9. *Optional:* Click **Add Exclusion** to define times when scheduled elements should not be executed.
10. *Optional:* Click **Add Definite Run** to define times when unscheduled executions should be executed.
11. Click **Save** to save your custom schedule.

Configuring Setup and Cleanup Executions

To define a test as a setup or cleanup test:

1. In the menu, click **Execution Planning** > **Details View**.
2. Click the execution plan for which you want to configure a setup or cleanup test.
3. Click the **Setup/Cleanup** tab.
   - To define a setup test, proceed with the following step.

- To define a cleanup test, proceed with step 5.

4. *Optional:* To define a setup test, click **Edit** in the **Setup Test** section. The **Edit Setup Test** dialog box displays.
5. To define a cleanup test, click **Edit** in the **Cleanup Test** section. The **Edit Cleanup Test** dialog box displays.
6. In the **Tests** tree, select the test that you want to configure.
7. Click **OK**.

The configured test displays in the corresponding section of the **Setup/Cleanup** page.

Adding Dependent Execution Plans

To add a dependent execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan that will act as the master execution plan.
3. Click the **Dependencies** tab.
4. Click **Add dependent Execution Plan**. The **Add dependent Execution Plan** dialog box displays.
5. From the **Condition** selection list, select the condition that is to trigger the dependent execution plan.
   - `Passed`
   - `Failed`
   - `Not Executed`
   - `Any`

   The `Any` status means that the dependent test execution will trigger no matter what the status of the previous test execution.
6. From the tree menu in the dialog box, select the execution plan that is to be dependent.
7. Specify where the dependent execution plan is to be deployed.

Option	Description
As specified in the dependent Execution Plan	Automated tests assigned to the dependent execution plan will be executed on the execution server specified for the dependent execution plan on the **Deployment** page. Manual tests assigned to the dependent execution plan will be assigned to the users specified for the dependent execution plan on the **Deployment** page.
Same as <selected execution plan's execution server>	Automated tests assigned to the dependent execution plan will be executed on the execution server specified for the master execution plan on the **Deployment** page. Manual tests assigned to the dependent execution plan will be assigned to the users specified for the master execution plan on the **Deployment** page.
Specific: Execution Server/ Manual Tester	Select a pre-configured execution server and/or a manual tester from the list boxes. Automated tests assigned to the dependent execution plan will be executed on the specified execution server. Manual tests assigned to the dependent execution plan will be assigned to the specified manual tester. If only a specific manual tester is defined and no server, only manual tests will be executed. If only a specific execution server is defined and no manual tester, only automated tests will be executed.

8. Click **OK** to create the dependency.

Note: Test Manager will not allow you to create cyclical execution dependencies. You can select conditions to fulfill for manual tests. For example if the selected condition is `Failed` and all manual tests passed, but some automated tests failed, only automated tests assigned to the dependent execution plan are executed.

Assigning Keywords to Execution Plans

To assign keywords to execution plans:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Deployment** tab.
4. In the **Execution Environment** section of the page, click **Edit**. The **Assign Keywords** dialog box displays. All keywords that have been defined for your execution environment are listed here.

   Note: The default reserved keywords for each execution server, `#<execution name>@<location name>`, are included in the list.
5. Select keywords in the **Select keywords** list box that reflect your execution environment requirements.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple keywords using standard browser multi-select functions.

   Tip: The **Enter keywords** text box is auto-complete enabled. When you enter alphanumeric characters into this field, the field is dynamically updated with an existing keyword that matches the entered characters. The text box is disabled when multiple keywords are selected in the **Select keywords** or **Assigned keywords** list boxes.

   Tip: If you only have a few execution servers and do not require hardware provisioning, it might be enough to use only the default, reserved keywords that are created for each execution server. In such cases, it is not necessary that you select additional keywords.
6. Click **>** to move the keyword into the **Assigned keywords** list box.

   Note: The execution servers that match the assigned keywords are listed below in the dynamically-updated **Matching execution servers** list box. This list updates each time you add or remove a keyword. Click on the name of an execution server in the list to access the execution servers in **Administration** > **Execution Servers** .
7. Click **OK** to save the keywords and close the **Assign Keywords** dialog box.

Starting Execution Plans

To run an execution plan independent of a schedule:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan, folder, or configuration suite that you want to execute.
3. Click **Run** on the toolbar. The **Run** dialog box displays.
4. Define which tests you want to execute in the **Run** dialog box.
5. If the execution plan does not contain pending manual tests, the **Go To Activities** dialog box displays. Click **Yes** to view the details of the execution plan runs in the **Activities** page, or click **No** if you want to remain on the current Web page.

   Note: Check the **Don't show this dialog again (during this login session)** check box if you do not want to be asked about switching to the **Activities** page again in the future. This setting will be discarded when you log out of Test Manager.

Viewing Test Run Details

To view the details of a test run:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan in the **Execution** tree.
3. Click the **Runs** tab.

4. Click the **Run ID** of the execution for which you want to see details. The **Test Run Results** dialog box displays.
5. Click the **Details** tab.

Managing Tests - Quick Start Task

To organize your tests, you will need to perform some or all of the following tasks.

Creating Tests

To create a new test:

1. In the menu, click **Tests** > **Details View** .
2. Select a container or folder node in the **Tests** tree where you want to insert a new test.
3. Click on the toolbar or right-click within the tree and choose **New Child Test**.

   A new test node is appended to the tree view, and the **New Test** dialog box opens.
4. Type a name and description for the test.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
5. Select the test type from the **Type** list box:
6. Click **Next** and proceed to the appropriate topic for the selected test type in *Configuring Test Properties*.

Note: The well-defined public API of Test Manager enables you to implement a proprietary solution that meets your automated test needs. Test Manager is open and extensible to any external tool that can be invoked from a Java implementation or through a command-line call.

Note: Throughout the test configuration process and across all test types, **Inherit from parent** check box options are provided where applicable, enabling you to accept settings of any existing parent entity.

Editing Tests

To edit a test:

1. In the menu, click **Tests** > **Details View** .
2. Select the test or the test package that you want to edit.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
3. Click **Edit** on the toolbar.

   You can also click the **Properties** tab and click **Edit** or right-click the test or test package and select **Edit**.

   The **Edit Test** dialog box displays.
4. Edit the name and description of the selected test.

   If the selected test is a test package, the **Update Package Structure on Result** check box is available. Check the **Update Package Structure on Result** check box if you want to update the structure of the test package according to the results of the test execution run.
5. Configure the properties of the test or the test package according to the test type as described in *Configuring Test Properties*.

Creating a Test Package

To create a new test package out of a third-party test:

1. Run the test once to create the `output.xml` file, which contains the structure of the test package.

For additional information on executing a test, see *Executing Individual Tests*.

2. In the menu, click **Tests** > **Details View** .
3. In the **Tests** tree, right-click the test and choose **Convert to Test Package**. The selected test is converted to a hierarchy representing the structure of the last execution result.

Creating Data-Driven Tests

To create a data-driven test:

1. In the menu, click **Tests** > **Details View** .
2. Create a new test.

   For information about creating a test, see *Creating Tests*.
3. Click the newly created test's **Properties** tab.

   Select the **Edit** icon to open the **Data-driven Properties** dialog box.
4. Scroll down to the **Data-driven Properties** section of the **Properties** tab and click . The **Data-driven Properties** dialog box opens.
5. Select a pre-configured data source from the **Data Source** list box.

   For information on configuring data sources, refer to the *SilkCentral Administration Module Help*.
6. Click **Next** to continue.
7. Select a data set from the **Data Set** list box.

   In the case of Excel data sources, this is a worksheet name. In the case of database data sources, this is a table name.
8. Check the **Each data row is a single test** check box to have each row in your data set considered to be a separate test, or do not check this check box to create a single test for all data rows of your data set.
9. *Optional:* You can enter a SQL query into the **Filter query** text box to filter your data set based on a SQL-syntax query.

   Note: Only simple `WHERE` clause queries are supported.
10. Check the **Enable data-driven properties** check box to enable data-driven functionality.
11. Click **Finish** to save your settings.

Note: Data-driven property settings are visible in the lower portion of each test's **Properties** page.

Note: To use the data-driven test functionality of Test Manager with SilkPerformer scripts, data sources with column names matching the corresponding SilkPerformer project attributes must be used in conjunction with `AttributeGet` methods.

Assigning Attributes to Tests

To assign an attribute to a test:

1. In the menu, click **Tests** > **Details View** .
2. Select the test to which you want to assign an attribute.
3. Click the **Attributes** tab.
4. Click **Add Attribute** to display the **Add Attributes** dialog box.
5. Click the plus symbol (+) in the **Add** column of the attribute that you want to assign. Based on the attribute type you have selected, an **Edit Attribute** dialog box displays, where you can specify which of the available attribute values you want to assign to the test.
6. Select attribute value and click **OK** to assign the attribute to the test.

Adding Predefined Parameters to Tests

To add a predefined parameter to a test:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test node to which you want to add a predefined parameter.
3. Click the **Parameters** tab.
4. Click **Add Predefined Parameter**.

   Note: For SilkPerformer tests, the **Add Predefined Parameter** button is only available for tests for which the **Project** property is already defined.

   The **Add Predefined Parameter** dialog box displays, which lists all of the project attributes that are available in the project file.
5. To add any of the listed parameters, click the corresponding add icon.
6. On the dialog box that displays, specify the actual value for the parameter.
7. Click **Save** to add the parameter to the active **Tests** tree node.

Creating Filters

To create a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Filters** tab to view the list of available filters.
3. Click **New Filter**. The **New Filter** dialog box displays.
4. Type a **Name** for the new filter.

   This name will be displayed in list boxes when the filter becomes available.
5. Select a **Category** for the new filter from the list box to make the filter available in a specific Test Manager area.

Category	Area
Requirement	Requirements
Test	Tests
Execution	Executions

6. Type a **Description** for the new filter.
7. *Optional:* Check the **Visible to other users** check box to allow other users to see the filter.
8. *Optional:* Check the **Editable by other users** check box to allow other users to edit the filter.

   Note: If **Visible to other users** and **Editable by other users** are checked, the filter is public. To delete private filters, you have to be the owner of the filter or you need the **Delete private filters of other users** permission.
9. Select a category of filter criteria from the **Selection criteria** list box. The available categories depend on the general filter category you have selected.

   Note: You can combine filters by selecting **Nested Test Filter** or **Nested Requirements Filter**. Selecting one of these categories allows you to include an existing filter in your new filter.
10. Select a **Property**, **Operator**, and **Value** for the filter from the respective list boxes.

    Property Available properties depend on the filter category that you have selected in the previous step. Defines the property for which you are defining a filter setting. If you have selected an attribute category, the property list includes custom attributes to query against.

Operator Specifies the filter operator. The operator depends on the property type you have selected. For example, if you have selected a property that is based on a string field type, the following operators are available:

Operator	Description
=	The string equals the defined value.
not	The string is different than the defined value.
contains	The string contains the defined value.
not contains	The string does not contain the defined value.

Value Enter the value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.

11. *Optional:* Click **More** if you want to add more than one filter category to the new filter. Repeat this procedure to define new categories.

    Note: If you define more than one filter category, you must define whether the categories need to be fulfilled in addition to the existing categories (AND relationship), or if the filter returns true when one of the filter categories is fulfilled (OR relationship). Select either **AND** or **OR** to define the filter category relationship. You cannot define nested AND, OR relationships.

12. *Optional:* To remove filter categories, click **Fewer**. This removes the last filter category.
13. Click **OK** to save the new filter, or click **Cancel** to abort the operation.

Assigning Requirements to Tests

To manually assign requirements to tests:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test to which you want to assign requirements.
3. Click the **Assigned Requirements** tab.

   Note: If you have created a requirements filter, you can select it from the filter list above the **Available Requirements** tree. To create a new requirements filter, navigate to **Requirements** > **Details View** and click on the toolbar.

   All requirements that are available for assignment are displayed in the **Available Requirements** tree.
4. Click to the left of a requirement or double-click on the requirement to assign it to the selected test.

   Note: Newly generated tests can automatically be assigned to the requirements from which they are generated by checking the **Assign newly generated Tests to Requirements** check box on the **Generate Tests from Requirements** dialog box. This is the default behavior.

Attaching Files to Test Elements

To attach a file to a test element:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a container, folder, or test.
3. Click the **Attachments** tab.
4. Click **Upload File**. The **Upload File** dialog box opens.
5. Click **Browse** to select a file from your local file system.
6. Enter a description for the attachment.
7. Click **Ok** to upload the attachment to the server and associate it with the selected element.

Note: Attaching files to a test element may not work in Mozilla Firefox. Firefox requires usage of three slashes, for example `file:///`, for a file link, while other browsers require only two, for example `file://`. Additionally, Firefox includes a security feature blocking links from remote files to local files and directories. For more information, see *http://kb.mozillazine.org/Firefox_:_Issues_:_Links_to_Local_Pages_Don't_Work*.

Attaching Links to Test Elements

To attach a link to a test element:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a container, folder, or test.
3. Click the **Attachments** tab.
4. Click **Attach Link**. The **Attach URL** dialog box opens.
5. Type the URL in the **URL** text box.
6. Type a description for the attached link.
7. Click **Attach URL** to associate the link with the selected element.

My Dashboard

The dashboard in Test Manager is your personal starting area. You can customize it by adding and arranging various panels, which give you the information you need for your daily work.

When you log in to Test Manager for the first time, the dashboard shows just the **Introduction** panel. On the top, your last login date and time is displayed. You can quickly **Continue working in your last visited area** by clicking this link.

Click the buttons on the top, to perform the following actions:

- Click **Add Panel** to add further panels to the dashboard.
- Click **Change Layout** to change the layout of the dashboard.
- Click **Reset Dashboard** to set the dashboard to the default view. All the panels you added and your customized settings are removed.

To arrange your panels, click on the header of a panel and drag it to the desired position.

Click the buttons in the header of a panel to perform the following actions:

- Click to remove the panel from the dashboard.
- Click to configure the settings of the panel.
- Click to save the content of the panel as a PDF.
- Click to print the content of the panel.

You need certain permissions to view or edit the content of the different panels.

Adding Panels to the Dashboard

To add a panel to the dashboard:

1. In the menu, click **Home** > **My Dashboard**.
2. Click **Add Panel** on the top left. The **Add Panel** dialog box appears, listing the available panels along with a short description for each panel.
3. Select a panel.
4. Click **OK**.
5. If required, select a project and a time span for the panel.

6. Click **OK.**

For most panels, you need to configure a project, as the panel will only show data for a certain project. You can add several panels to also show appropriate data for other projects. Click in the header of a panel to configure a different project. You need certain permissions to view or edit the content of the different panels.

Dashboard Panels

For most panels, you need to configure a project because the panel will only show data for a certain project. Click in the header of a panel to configure a project. You can add several panels to also show appropriate data for other projects. You need certain permissions to see the content of the different panels.

You can add the following panels to your dashboard:

Panel	Description
Custom Information	This panel shows customizable information about a certain project. Users with the **Manage projects** permission can add content like text, images or hyperlinks to the panel. Click **Edit Content** to open the HTML editor. Users without the **Manage projects** permission will just see the content. The panel can be used to show news, descriptions or any other kind of information relevant to the project.
Introduction	This panel gives you a quick introduction to Test Manager. Click the links under **What's New?** to get additional information about the latest features of Test Manager. You can also join the **Micro Focus Community** or contact the **Micro Focus SupportLine**.
Issue Life Cycle	This panel shows the number of **Open**, **Fixed**, **Verified**, **Closed** and **Deferred** issues for a defined issue tracking profile. Select a product in the **Product** list, and a time span of 30, 90, 180 or 360 days in the **Time Span** list. To define an issue tracking profile, click **Issues** in the menu, click **Issue Tracking Integrations** and click **New Profile**. The panel will only show data, if there is a profile defined.
Issues Created per Tester	This panel shows the number of issues created by testers as a bar graph. When printed or downloaded as a PDF, the numbers will be shown in a table as well. Select a time span between 1 and 4 weeks from the list.
Manual Tests Assigned to Me	This panel shows all tests from a certain project that were assigned to you. Click to quickly continue working on that test. Click to open the **Manual Testing Client**. Click to get detailed information about the **Test**, the **Execution Plan** or the **Execution Plan Parent**.
Quality Goal Progress	This panel shows a graph with two bars for each value of the selected quality goal. The upper bar shows the number of tests that must be executed to meet the quality goal. The lower bar shows the number of the **Passed**, **Failed**, **Not Executed**, and **N/A** (not available) tests with that quality goal value. If the lower bar is shorter than the upper bar, too few tests are assigned to execution plans.

Panel	Description
	This means that the quality goal cannot be reached. Select a quality goal from the list.
Requirements Coverage Status	This panel shows if the requirements are covered by manual or automated tests. Click to go to the **Requirements Document View**.
Testing Cycle Progress	This panel shows a burn-up chart of **Passed**, **Failed**, **In Progress**, and **Not Yet Executed** tests for the defined testing cycle. Select a testing cycle from the list.
Testing Cycle Result Summary	This panel shows the status of the selected testing cycle and the progress status of the individual testers that are assigned to the testing cycle. The bars show in the following order: the number of **Tests failed**, **Tests remaining**, and **Tests passed**. The tester with the most failed tests displays on the top of the list. The numbers to the right of the bars show the **Tests completed** (left number) and the **Tests assigned** (right number). **No specific tester** is treated and ordered like all other testers. Click to go to the **Test Assignment** view of the **Manual Execution Planning** page.
Testing Progress Across Testing Cycles	This panel shows the time in hours for the **Planned** tests (visualized by a line), and for the **In Progress** and the **Completed** tests (both visualized by stacked areas). The panel helps a test manager to predict if the testing team will be able to finish the tests in time. The start date of the chart is the start date of the first testing cycle. The end date of the chart is the end date of the last testing cycle. The panel does not show how long it actually took the testing team to execute the tests.
Volatile Tests	This panel shows how often tests changed their status within an execution plan. Tests without status changes and test packages do not display in the panel. Tests display according to the selected project and time span. The panel makes test managers aware of such volatile tests, which indicate either an unstable test or a tested area where bugs occur frequently.

Dashboard Panel Permissions

To view or edit the content of a certain dashboard panel, you need the following permissions:

Panel	Permission
Custom Information	**Manage projects** required for editing. No permission required for viewing.
Introduction	No permission required.
Issue Life Cycle	**View project settings**.
Issues Created per Tester	**View tests and executions**.
Manual Tests Assigned to Me	**Manage tests and executions**.
Quality Goal Progress	**View quality goals**.
Requirements Coverage Status	**View requirements**.
Testing Cycle Progress	**View testing cycles and configurations**.

Panel	Permission
Testing Cycle Result Summary	**View testing cycles and configurations**.
Testing Progress Across Testing Cycles	**View testing cycles and configurations**.
Volatile Tests	**View tests and executions**.

Projects

This section explains how to manage projects in Test Manager. The **Projects** area offers a high-level test-manager's view of all projects in your Test Manager installation, including a list of the most recent projects you have set to active, and enables you to move between projects. The area also enables you to manage projects and their settings, see high-level project status details, and view current execution statistics.

Build Information

Build information files contain version and build information that is used for execution runs. The build information files are typically stored and searched for on the execution server that is executing the execution run. If a build information file is not found on the execution server, the file is searched for on the application server. This behavior is beneficial when you have multiple execution servers and use a single build information file across all execution servers. You need to maintain only a single build information file on the application server. The build information files for automated tests that have no execution server assigned, and for all manual tests, are immediately searched for on the application server.

Test Manager is able to match up test results with build information and display test results for specific build numbers.

Build Information Updates

Build information files must be created and configured manually. Whenever a new build becomes available for testing on an execution server, update the build information to reflect the new build number.

You can update the build information in two different ways:

- Manually, by editing the files each time a new build is installed.
- Automatically, if you are using an automated build update process to update the build information file, for example through a VB Script.

Creating Build Information Files

To create a build information file:

1. On both the application and execution servers, navigate to: `C:\Documents and Settings\All Users\Application Data\SilkCentral\BuildInfos`.
2. Create a build info file for your project based on the template file `BuildInfoExample.xml`. The template file contains the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<ProjectBuildInfo>
  <BuildEntryList>
       <BuildEntry name="Demo Product">
            <Version>3.1</Version>
            <BuildNr>350</BuildNr>
    </BuildEntry>
       <BuildEntry name="Product2">
            <Version>4.2</Version>
            <BuildNr>613</BuildNr>
    </BuildEntry>
```

```
  </BuildEntryList>
</ProjectBuildInfo>
```

Note: To improve the structure of build information files, an element called `BuildEntryList` which contains a list of `BuildEntry` elements has been created. `BuildEntry` tags refer to specific products that are defined by the name attribute of `BuildEntry` elements.

3. Modify the file content to fit your environment.

`Version`	Used on both application and execution servers. The number of the version that is currently available for testing. The number is not necessarily the same for each execution server.
`BuildNr`	Used on both application and execution servers. The number of the build that is currently available for testing. The number is not necessarily the same for each execution server.

4. Distribute the build information file to the execution servers: `C:\Documents and Settings\All Users\Application Data\SilkCentral\BuildInfos`.

Note: When stored on both the application server and execution servers, build information files must have the exact same name.

5. Once you have created the build information files on the application server and each execution server, you must specify the file name in the settings of the corresponding project. Click **Projects:<Project Name>** > **Projects List** to view the list of projects assigned to you. Select the project to which you want to link the build information.

Note: This must be done before the scheduling of any tests for the project. Otherwise previously scheduled tests will not be updated.

6. Click the **Project Settings** tab.
7. Click **Edit** to edit the project settings of the selected project. The **Edit Project Settings** dialog box opens.
8. Specify the name of the previously created XML file in the **Build information file name** text box.
9. Click **OK** to update the information. With all future test executions, Test Manager will read build information from the corresponding file and match test results with that information.

Comparing a Project with a Baseline

For baselined projects, you can use the *Baseline Comparison* report to see all the changes to the tests in the project since the moment the project was baselined.

To view the report:

1. In the menu, click **Projects** > **Project List**.
2. Use **CTRL + CLICK** to select the project and the baseline of the project that you want to compare.

Tip: For easier selection of the baselines, don't click on the name of the baseline. Click into another column instead.

3. Right click on the selection.
4. Select **Reports** > **Baseline Comparison** .

Note: The *Baseline Comparison* report must be executed once to display in the context-menu. If the report is grayed out, execute the report once in the **Reports** area.

Selecting Projects

To select a project:

1. In the menu, click **Projects** > **Project List**.
2. Click the name of the project to select it.

Note: You can only select active projects.

Project List

Project:<Project Name> > **Project List**

The **Project List** page alphabetically displays all the projects and project baselines associated with your Test Manager installation. The page supports filtering, but does not support sorting. The displayed projects are sorted by name. Inactive projects and inactive project baselines are grayed out.

Project baselines are sorted by their date of creation, marked with a special icon, and shown as subnodes of the original project. For more information on baselines, refer to the *SilkCentral Administration Module Help*.

The following columns are displayed for each project and project baseline:

Column	Description
Actions	• Delete • Edit • Copy project • Baseline project
Project	Name of the project. Click to set the project as the active project.
Project ID	The identifier of the project.
Status	`Active` or `Inactive`. Click to switch the status.
Description	Project description.
Created On	Date and time the project was created.
Created By	User who created the project.
Changed On	Date and time the project was last changed.
Changed By	User who last changed the project.

Note: Only your system administrator has rights to set up new projects.

Switching to a Recently-Accessed Project

You can quickly switch between the five Test Manager projects that you have most recently accessed.

To open one of the five projects you have most recently opened:

1. In the menu, click **Projects**.
2. In the **Recent Projects** list, click the name of the project that you want to access. You can continue to work with the selected project.

Settings Configuration

This section explains how to configure settings in Test Manager.

If you have SuperUser, Administrator, or Project Manager privileges, you can specify project-wide settings for Test Manager projects. Once global project settings are defined, they are available to all users who have access to those projects. Global project settings include the definition of filters, attributes, external product integrations, change notifications, build information, source files, file extensions, and more.

Project Settings

Project settings are settings that are specific to a project, for example the project release date.

Configuring Project Settings

To customize the project settings:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Project Settings** tab to view the current settings. The **Project Settings** page opens.
3. Click **Edit** to modify the current project settings.
4. The **Edit Project Settings** dialog box displays. You can specify the following information:

Build Information File Name	Build information files contain project information, including build number, build log location, error log location, and build location. Enter the name of your project's build information file in this text box. All test executions will read the build information from this specified file.
Project Release Date	Specify the planned release date for your project.
File Extensions to ignore in Results	Specify result file types or other file types that should not be saved as results for test executions.

   Note: File extensions must be separated by commas, for example, xlg, *_, res. Changes made in the **Build Information File Name** and **File Extensions to ignore in Results** fields will not effect scheduled tests. To redistribute tasks to execution servers, you must reschedule the tests, or disconnect from and reconnect to the database.
5. Click **Save** to save your project settings.

Project Settings Page

Project:<Project Name> > **Project Settings** > **Project Settings**

The **Project Settings** page lists the following high-level details about the active project:

Item	Description
Build information file name	Build information files contain project information, including build number, build log location, error log location, and build location. Enter the name of the active project's build information file in this field. All test executions will read the build information from this specified file.
Project release date	Scheduled release date of the active project in the format `MM/DD/YYYY`.
File extensions to ignore in results	Result file types or other file types that should not be saved as results for test executions.

Filters

Filters provide an efficient means of finding exactly the information you need, while excluding extraneous detail. Filters highlight only those elements that are relevant to your needs, and enable you to quickly sort through requirements, test elements, and execution plans. By defining global filters, you can create complex filter criteria that are available throughout Test Manager without defining filter criteria each time you need to filter a list.

Based on your needs, you can create new filters, edit existing filters, select filters, delete filters, or turn filtering off at the project level. Projects do not contain default filters. You can access and edit filters from the toolbars in the Test Manager units and from the **Project Settings** unit.

Note: Filters are not applied to reports. The **Recent Changes** filter enables you to view project-wide changes and additions that other users have made to tests since your last change acknowledgement. The **Show Changes/Show All** toggle button and the **Acknowledge** button in the **Tests** area help you to find out what changes other users have made. Your system administrator can configure email notifications that alert you to changes that are made to test settings. Email alerts include links that take you directly to a view of recent changes.

Creating Filters

To create a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Filters** tab to view the list of available filters.
3. Click **New Filter**. The **New Filter** dialog box displays.
4. Type a **Name** for the new filter.

   This name will be displayed in list boxes when the filter becomes available.
5. Select a **Category** for the new filter from the list box to make the filter available in a specific Test Manager area.

Category	Area
Requirement	Requirements
Test	Tests
Execution	Executions

6. Type a **Description** for the new filter.
7. *Optional:* Check the **Visible to other users** check box to allow other users to see the filter.
8. *Optional:* Check the **Editable by other users** check box to allow other users to edit the filter.

   Note: If **Visible to other users** and **Editable by other users** are checked, the filter is public. To delete private filters, you have to be the owner of the filter or you need the **Delete private filters of other users** permission.
9. Select a category of filter criteria from the **Selection criteria** list box. The available categories depend on the general filter category you have selected.

   Note: You can combine filters by selecting **Nested Test Filter** or **Nested Requirements Filter**. Selecting one of these categories allows you to include an existing filter in your new filter.
10. Select a **Property**, **Operator**, and **Value** for the filter from the respective list boxes.

    Property Available properties depend on the filter category that you have selected in the previous step. Defines the property for which you are defining a filter setting. If you have selected an attribute category, the property list includes custom attributes to query against.

    Operator Specifies the filter operator. The operator depends on the property type you have selected. For example, if you have selected a property that is based on a string field type, the following operators are available:

Operator	Description
=	The string equals the defined value.
not	The string is different than the defined value.

Operator	Description
contains	The string contains the defined value.
not contains	The string does not contain the defined value.

Value Enter the value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.

11. *Optional:* Click **More** if you want to add more than one filter category to the new filter. Repeat this procedure to define new categories.

Note: If you define more than one filter category, you must define whether the categories need to be fulfilled in addition to the existing categories (AND relationship), or if the filter returns true when one of the filter categories is fulfilled (OR relationship). Select either **AND** or **OR** to define the filter category relationship. You cannot define nested AND, OR relationships.

12. *Optional:* To remove filter categories, click **Fewer**. This removes the last filter category.
13. Click **OK** to save the new filter, or click **Cancel** to abort the operation.

Editing Filters

To edit a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Filters** tab to view the list of available filters.
3. Click the name of the filter you want to edit. The **Edit Filter** dialog box displays.
4. Edit the **Name** and **Description** of the filter.
5. Select a category of filter criteria from the **Selection criteria** list box. The available categories depend on the general filter category you have selected.

Note: You can combine filters by selecting **Nested Test Filter** or **Nested Requirements Filter**. Selecting one of these categories allows you to include an existing filter in your new filter.

6. Select a **Property**, **Operator**, and **Value** for the filter from the respective list boxes.

Property Available properties depend on the filter category that you have selected in the previous step. Defines the property for which you are defining a filter setting. If you have selected an attribute category, the property list includes custom attributes to query against.

Operator Specifies the filter operator. The operator depends on the property type you have selected. For example, if you have selected a property that is based on a string field type, the following operators are available:

Operator	Description
=	The string equals the defined value.
not	The string is different than the defined value.
contains	The string contains the defined value.
not contains	The string does not contain the defined value.

Value Enter the value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.

7. Click **OK** to save the edited filter definition.

Deleting Filters

To delete a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Filters** tab to view the list of available filters.
3. Click **Delete Filter** for the filter that you want to delete. A confirmation dialog box displays, asking you you to confirm the deletion.
4. Click **Yes** to delete the selected filter or **No** to abort the operation. If you click **Yes**, you will be returned to the filters list and the deleted filter is no longer displayed.

Filters Page

Project:<Project Name> > **Project Settings** > **Filters**

The **Filters** page lists the filters that are available to the active project. For each filter, the **Filters** page displays the following columns:

Item	Description
Actions	Actions that can be performed on the filter. Currently only **Delete**.
Name	Name of the filter.
Type	Filter category. Requirement, test, or execution.
Created On	When the filter was created.
Created By	User who created the filter.
Changed On	When the filter was most recently modified.
Changed By	User who most recently modified the filter.

Attributes

You can use custom attributes to customize the information for tests in the **Tests** area. While some attributes are made available by the integrated functionality of Test Manager, such as priority, components, and platforms, you may want to define custom attributes to categorize tests to your needs, or to make tests compatible with specific test cases.

Creating Custom Attributes

To create a custom attribute:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Attributes** tab to view the list of current attributes.
3. Click **New Attribute**. The **New Attribute** dialog box displays.
4. Type a **Name** for the new attribute. This name will be displayed in list boxes when the attribute becomes available for use.
5. Type a **Description** for the new attribute.
6. Select the attribute **Type**. See the *Test Attribute and Requirement Property Types* topic for descriptions of each type of attribute.
7. Click **OK**.

Test Attribute and Requirement Property Types

Test Manager supports the following types of test attributes and requirement properties:

Date The **Date** type supports date values.

List The **List** type supports single select or multi-select lists. It has the following controls:

- **New Item**: Allows you to create entries in the list. Click to show the **New Item** dialog box. Enter the **Name** and **Numeric Weight** for the list item and click **OK**.

  Note: The value of the **Numeric Weight** field is used during filtering when the filter criteria uses any of the following operators: **>**, **>=**, **<**, or **<=**. When the filter criteria uses the **=** or **not** operators, the value from the **Name** field is used for comparisons.
- **Allow Multiselect**: click this check box to allow users to select more than one value from the list.
- **Actions**: use the icons in this row to delete, to move up, to move down or to edit each item in the list.

Number The **Number** type supports integer or decimal numeric values. It has the following fields:

- **Minimum value**: represents the smallest value that you can enter into the field.
- **Maximum value**: represents the largest value that you can enter into the field.
- **Decimal places**: used to define how many values to the right of the decimal point are valid. Select `0` for an integer value or `1`, `2`, `3`, or `4` for a decimal value.

Note: Custom attributes and requirement properties display their current value if an update is made to their definition (for example, changes to **Minimum value** or **Maximum value**). In order to apply the new definition, open and save the asset.

Text The **Text** type supports any alphanumeric value. It has the following fields:

- **Maximum length**: represents the maximum number of characters that can be entered into the field.

Editing Custom Attributes

To edit a custom attribute:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Attributes** tab to view the list of current attributes.
3. Click the name of the attribute that you want to edit. The **Edit Attribute** dialog box displays.
4. Edit the **Name** of the attribute.

   When the attribute is available for use, the name will display in list boxes for filters and tests. Attributes can be used in global filters for filtering by test attributes and can be applied to tests.
5. Edit the **Description** of the attribute.
6. See the *Test Attribute and Requirement Property Types* topic for descriptions of each type of attribute. You have different editing options depending on the attribute data type.
7. Click **OK**.

   You are returned to the **Attributes** list.

Deleting Custom Attributes

To delete a custom attribute:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Attributes** tab to view the list of current attributes.
3. Before you can delete an attribute, you must first deactivate it. In the **Status** column, click the **Active** link or icon and then click **Yes** on the confirmation dialog box to deactivate the attribute.
4. Once the attribute is inactive, click **Delete Attribute** to remove it. A confirmation dialog box displays, asking you to confirm the deletion.
5. Click **Yes** to remove the selected attribute; or click **No** to abort the operation. If you click **Yes** you will be returned to the **Attributes** page, where the removed attribute is no longer displayed.
6. If an error displays, ensure that the selected attribute is not applied to any tests or used in any global filters. You can only delete unused attributes.

Attributes Page

Project:<Project Name> > **Project Settings** > **Attributes**

The **Attributes** page lists the attributes that have been created for the current project. For each attribute, the **Attributes** page displays the following columns:

Column	Description
Actions	Available actions that can be performed on the attribute.
Name	Name of the attribute. When the attribute is available for use, the name will display in list boxes for filters and tests. Attributes can be used in global filters for filtering by test attributes and can be applied to tests. Click the **Edit** icon to edit the name of the attribute.
Type	Attribute type. See the *Test Attribute and Requirement Property Types* topic for descriptions of each type of attribute.
Status	Status of the attribute, active or inactive.
Column	The column name of the attribute in the `LQM Reporting` table. Use this column name to query the selected attribute within the `LQM Reporting` table. See the database model documentation for detailed information.
Created On	When the attribute was created.
Created By	User who created the attribute.
Changed On	When the attribute was most recently modified.
Changed By	User who most recently modified the attribute.

Requirement Properties

The following sections describe the two types of requirement properties that are available.

Requirement Properties Page

Project:<Project Name> > **Project Settings** > **Requirement Properties**

The **Requirement Properties** page lists the custom requirement properties and the calculated property that are available for the active project.

Calculated Property

Column	Description
Name	Name of the calculated property.
Description	Description of the calculated property.
Formula	Contains the actual formula of the calculated property.
Classifications	Lists the classifications for the calculated property.
Create/Edit Calculated Property	Click to open the **Edit Calculated Property** dialog box to modify your calculated property. If no property exists, the button is named **Create Calculated Property**.
Delete Calculated Property	Click to delete the calculated property.

Custom Properties

Column	Description
Actions	Available actions that can be performed on the custom property.
Name	Name of the requirement property. Click the **Edit** icon to edit the name of the requirement property.
Type	Property type. See the *Test Attribute and Requirement Property Types* topic for descriptions of each type of attribute.
Created On	When the property was created.
Created By	User who created the property.
Changed On	When the property was most recently modified.
Changed By	User who most recently modified the property.

Calculated Properties

Calculated properties are used to create a single value based on a user-defined formula derived from the values of internal or custom properties. This value is then put into a labelled range called a *classification.* The goal of a calculated property is to be able to view the classification **Name** in grids. The value of a calculated property is shown everywhere where the values of custom properties are shown (Document view, for example). Calculated properties can also be used in filters and quality goals like other internal or custom requirement properties.

Every calculated property has a default classification without a lower bound. All other classifications must have a lower bound. The classification allows you to set a name and range for the value of the calculated property. The name of the classification is how the value will display in the grids. If one or more property values used in the formula are not set, or evaluating the formula on a particular requirement fails (for example, division by zero), `---` will appear as the value of the calculated property.

You can use any internal or custom property from within a project in the formula for a calculated property. The internal or custom property must be of type `Number`, `Text`, or `List` with single selection. You can use the following operators in the formula: `+ - * / ( )`

Note: You can only have a single calculated property per project.

Creating a Calculated Property

Calculated properties are used to create a property that is based on a range of values. This procedure will walk you through creating a calculated property named `Custom Risk` with three classifications: `Low`,

Medium, and High. We will use the internal property **Risk** and a custom property named **Impact** of type Number (you should create this one in advance).

1. Click **Project:<Project Name>** > **Project Settings** > **Requirement Properties** .
2. Click **Create Calculated Property**.
3. In the **Name** field, type Custom Risk.
4. In the **Description** field, type Calculated property using Risk and Impact (L, M, H).
5. Select Impact from the **Insert Property** list. ${Impact} is added to the **Formula** field.
6. Select * from the **Insert Operator** list. ${Impact}* is added to the **Formula** field.
7. Select Risk from the **Insert Property** list. ${Impact}*${Risk} is added to the **Formula** field.
8. On the default row of the **Classification** grid, rename the entry to Low. Since all calculated properties must contain a default classification without a lower bound, this one will be used.
9. Click **Add Classification**. A new row is added to the **Classification** grid.
10. Name the classification Medium and add a **Lower Bound** of 10.
11. Click **Add Classification**. A new row is added to the **Classification** grid.
12. Name the classification High and add a **Lower Bound** of 20.
13. Click **OK**.

A calculated property is now created that will be populated with Low, Medium, or High when both the **Impact** and **Risk** properties are set for a requirement.

Calculated Property Page

The **Calculated Property** page is used to add or to edit a calculated property to a project. To access this page click **Project:<Project Name>** > **Project Settings** > **Requirement Properties** and then click **Edit Calculated Property** or **Create Calculated Property** depending on whether or not a calculated property exists or not.

Name	Contains the name of your calculated property.
Description	Enter a meaningful description of your calculated property perhaps referencing the formula or classification.
Formula	Contains the formula for your calculated property. You can manually type in the formula or use the **Insert Property** and **Insert Operator** lists to do this. If you manually type in the formula, the field names should be contained within the following structure: ${}. Operators go between this structure.
Insert Property	Place your cursor where you want the property to appear in the **Formula** field and select one of the system or custom properties from the list to add it.
Insert Operator	Place your cursor where you want the operator to appear in the **Formula** field and select an operator.
Classifications Grid	The **Classifications** grid is used to set varying ranges for the values of your calculated properties. Click **Add Classification** to create a new row in the grid. Use the **Name** field to identify the classification and the **Lower Bound** field to determine the lowest value for calculated property. The important thing to know about classifications is that the **Name** of the classification is what will appear in grids, not the values of calculated properties. For example if your classification **Name** is Medium with a **Lower Bound** of 5 and your calculated property value is 7, as long as your next classification lower bound value is greater than 7, the calculated value will appear as Medium.

Internal Property Values

The values of several internal properties can be used in the formulas for calculated properties. The values for supported properties are noted below.

Risk

Value Name	Internal Value
Low	1
Medium	2
High	3
Critical	4

Priority

Value Name	Internal Value
Low	1
Medium	2
High	3
Critical	4

Reviewed

Value Name	Internal Value
No	0
Yes	1

Custom Properties

You can add custom property fields across all requirements in the selected project in **Requirements** > **Requirement Properties**. Custom properties can subsequently be edited alongside the default properties on the **Edit Requirements** dialog box. Custom properties are displayed in **Requirements** > **Details View** > **Properties**.

Creating a Custom Requirement Property

To create a new custom requirement property:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Requirement Properties** tab.
3. Click **New Requirement Property** to display the **New Requirement Property** dialog box.
4. Type a name for the new property in the **Name** field.
5. Select the data **Type** of the new property from the **Type** list. See the *Test Attribute and Requirement Property Types* topic for descriptions of each type of attribute.
6. Click **OK**.

Your custom property is available to all requirements in the active project.

Test Attribute and Requirement Property Types

Test Manager supports the following types of test attributes and requirement properties:

Date The **Date** type supports date values.

List The **List** type supports single select or multi-select lists. It has the following controls:

- **New Item**: Allows you to create entries in the list. Click to show the **New Item** dialog box. Enter the **Name** and **Numeric Weight** for the list item and click **OK**.

  Note: The value of the **Numeric Weight** field is used during filtering when the filter criteria uses any of the following operators: **>**, **>=**, **<**, or **<=**. When the filter criteria uses the **=** or **not** operators, the value from the **Name** field is used for comparisons.
- **Allow Multiselect**: click this check box to allow users to select more than one value from the list.
- **Actions**: use the icons in this row to delete, to move up, to move down or to edit each item in the list.

Number The **Number** type supports integer or decimal numeric values. It has the following fields:

- **Minimum value**: represents the smallest value that you can enter into the field.
- **Maximum value**: represents the largest value that you can enter into the field.
- **Decimal places**: used to define how many values to the right of the decimal point are valid. Select `0` for an integer value or `1`, `2`, `3`, or `4` for a decimal value.

Note: Custom attributes and requirement properties display their current value if an update is made to their definition (for example, changes to **Minimum value** or **Maximum value**). In order to apply the new definition, open and save the asset.

Text The **Text** type supports any alphanumeric value. It has the following fields:

- **Maximum length**: represents the maximum number of characters that can be entered into the field.

Editing Custom Requirement Properties

To edit a previously created custom requirement property:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Requirement Properties** tab.
3. Click the name of the property you want to edit. The **Edit Requirement Property** dialog box displays.
4. Edit the name of the property in the **Name** field.

   When the property is available for use, the name will display in lists for filters and requirements. Properties can be used in global filters for filtering by requirement properties and can be applied to requirements.
5. Edit the **Description** of the property.
6. See the *Test Attribute and Requirement Property Types* topic for descriptions of each type of requirement property. You have different editing options depending on the property data type.
7. Click **OK** to save your changes.

Deleting Custom Requirement Properties

To delete a previously created custom requirement property:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Requirement Properties** tab.
3. Click ✕. A confirmation dialog box appears.
4. Click **Yes** to confirm the deletion.

Step Properties

Use step properties to extend a manual test step with additional properties. Add a custom step property to a project in **Projects:<Project Name>** > **Project Settings** to add the property to all manual test steps in the project. Add a custom step property to a library in the **Libraries** tree to add a custom step property to all manual test steps in the library. Step properties can subsequently be edited alongside the default properties on the **Edit Manual Test Step** dialog box.

Creating Step Properties

To create a new step property:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Step Properties** tab.
3. Click **New Property**. The **New Step Property** dialog box opens.
4. Enter a name for the new property in the **Name** text box.

   Note: Step property fields are always declared as type string.
5. Click **OK** to make your property available to all manual test steps in the selected Test Manager project.

   Note: To create a step property for a library, select the library node in the **Libraries** tree and continue with the second step.

Editing Step Properties

To edit a previously created step property

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Step Properties** tab.
3. Click on the name of the property that you want to edit. The **Edit Step Property** dialog opens.
4. Edit the name of the property in the **Name** text box.
5. Click **OK** to save your changes, or click **Cancel** to abort the operation without saving.

Deleting Step Properties

To delete a previously created step property

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Step Properties** tab.

3. Click **Delete Property** in the **Actions** column of the property that you want to delete. A confirmation dialog box displays, asking you to confirm the deletion.
4. Click **Yes** to complete the operation, or **No** to abort.

Step Properties Page

Project:<Project Name> > **Project Settings** > **Step Properties**

The **Step Properties** page lists all properties that can be populated into manual test steps across the active project. For each custom step property, the **Step Properties** page displays the following columns:

Column	Description
Name	Name of the step property.
Actions	The following actions can be performed on a property: • **Delete Property** • **Move Up** • **Move Down**

Change Notification

Test Manager can notify you by email when requirements or tests are changed by other users. Each user has the option of activating change-notification. Once notification has been enabled, you can view and acknowledge changes that have occurred since your last acknowledgment. To avoid numerous notifications, only a single email alert is sent to you when a change is made, regardless of how many changes other users may have made since your last acknowledgment. Email alerts include links that take you directly to a view of recent changes.

Before you can activate change notification for requirements or tests, you must configure your email address in the user settings of Test Manager. Refer to the SilkCentral Administration Module Help for details.

Note: Change notification only works if an email server has been configured by your administrator. If change notification has not been enabled, please contact your SilkCentral administrator.

Enabling Change Notification

To enable change notification:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Notifications** tab.
3. Click **Configure Notification** to open the **Configure Change Notification** dialog box.
4. If you want to be notified by email when changes are made to requirements in the currently selected project, check the **Changes on Requirements** check box.
5. If you want to be notified by email when changes are made to tests within the currently selected project, check the **Changes on Tests** check box.
6. Click **OK** to save the notification settings, or click **Cancel** to abort the operation without saving changes.

You will be notified by email about changes for which you have activated notification.

Disabling Change Notification

To disable change notification:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Notifications** tab.
3. Click **Configure Notification** to open the **Configure Change Notification** dialog box.
4. If you do not want to be notified by email when changes are made to requirements in the currently selected project, uncheck the **Changes on Requirements** check box.
5. If you do not want to be notified by email when changes are made to tests in the currently selected project, uncheck the **Changes on Tests** check box.
6. Click **OK** to save the notification settings or click **Cancel** to abort the operation without saving changes.

Changes Triggering Change Notification

When you have activated change notification to inform you of changes that are made to requirements or tests an email alert is sent to you, following your logout. The email alert is sent to you if one or more of the following settings are changed:

Area	Changes
Requirements	• A requirement is created or deleted. • The name or description of a requirement is edited. • A system property is edited. • A requirement is set as obsolete. • A requirement is recovered. • A test is assigned to or removed from a requirement. • A custom property of a requirement is created, edited, or deleted.
Tests	• A container is created or edited. • A product is edited. • A source control profile is edited. • "Clear working folder" is edited. • The root node is edited. • The custom data directory is edited. • The include directory is edited. • The hidden test properties are edited. • The SilkTest Classic interface is edited. • A test folder is created, edited, or deleted. • A test is created, edited, or deleted. • The planned time is edited. • A test step is added or edited.

Notifications Page

Project:<Project Name> > **Project Settings** > **Notifications**

The **Notifications** page lists the notification events for the active project. For each notification event, the **Notifications** page displays the following columns:

Column	Description
Notification Events	Name of the notification event that has been set up for the active project.
Status	Status of the notification event. Active or inactive.

Click the **Configure...** button in each section to edit the values.

Email

When an email notification event is activated, a notification email is sent to the user that activated the event, the first time one of the specified settings is changed.

Note: You must have specified an email address to be able to receive email notifications. To specify an email address, refer to the *SilkCentral Administration Module Help*.

Requirement Flagging

You can configure notifications differently for internal and external requirements. All requirement flagging is enabled by default. For more information, see the *Flags* topic.

External Requirements Management Tools

This section explains how to work with external requirements management tools.

Requirements Management Configuration

Integrating external requirements-management enables you to coordinate the requirements-management features of Test Manager with other tools you may already be working with. Integration is configured on the **Requirements Management** page in **Project:<Project Name>** > **Project Settings**. The page is divided into a separate section for each installed plug-in. Initially, the view is divided into sections, one for each pre-installed requirements-management tool.

For information on the supported versions, refer to the *Test Manager Release Notes*. You can download add-ins for Rational RequisitePro and Telelogic DOORS from the **Tools** unit in **Help**. These add-ins can be installed on appropriate server and client computers. For details, refer to the `ReadMe` files that are included in the downloadable archives.

Test Manager supports integration with external requirements-management systems (RMS) through its open interface. Creating a plug-in and integrating it into Test Manager allows integrating any RMS. Refer to the *Test Manager API Help* for information about the interfaces that enable proper integration of external RMS.

Before you can configure CaliberRM integration, you must install the CaliberRM client on the SilkCentral application server and on the front-end server. Additionally, make sure that MPX support is enabled in CaliberRM.

Before you can configure RequisitePro integration, you must install the IBM Rational RequisitePro client on the SilkCentral front-end server.

Before you can configure DOORS integration, you must install the Telelogic DOORS client on the SilkCentral front-end server.

- The add-in for RequisitePro enhances the RequisitePro menu with an entry providing a link to the Test Manager front-end server's project selection.
- The Add-In for DOORS enables Test Manager to communicate with DOORS. This add-in must be installed on the DOORS client on the SilkCentral front-end server.

Note: Configuring integration with CaliberRM requires the definition of CaliberRM login credentials. Whenever requirements are synchronized between Test Manager and CaliberRM, these credentials are used to login to Test Manager, thus checking out a Test Manager license. The license is set free as soon as the synchronization process has completed. We recommend creating a dedicated Test Manager user for synchronization purposes, which should be used by all Test Manager requirements integrations. This ensures that only a single CaliberRM license is used for the process of synchronization.

Requirements Management Page

Project:<Project Name> > **Project Settings** > **Requirements Management**

The **Requirements Management** page lists the requirements-management integrations that have been configured for the current project.

Note: If an integration has not been enabled, you will only see the **Status** property.

CaliberRM Integration

This section lists details related to the integration of the CaliberRM requirements management system.

Item	Description
Status	Status of integration, enabled or disabled.
Hostname	Machine where the external server is installed.
Username	Credential for the requirements management server.
Project Name	External project with which the Test Manager project is integrated.
Baseline	Baseline of the external project with which the Test Manager project is integrated.
Requirement Types	Requirement types within the project that are integrated.
Schedule	Schedule to automatically synchronize requirements with CaliberRM.
Create Requirements	Indicates whether or not the **Enable creation of unassigned requirements** option is active. Enables creation and editing of unmapped requirements in Test Manager projects that are configured for integration with CaliberRM.
Upload Requirements	Indicates whether or not the **Enable upload of requirements to CaliberRM** option is active. Enables the upload of unmapped or unassigned requirements from Test Manager to CaliberRM. This allows you to upload additional previously unmapped requirement trees to CaliberRM and then have those requirements mapped within Test Manager. When this option is enabled, the **Map Requirement** button in **Requirements** > **Properties** becomes enabled, enabling configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
Property Mappings	Lists any external to internal property mappings that have been defined between the internal and external requirements management systems.
Actions	• **Edit Configuration** • **Edit Property Mapping** • **Edit Schedule** • **Edit Notification** • **Disable Configuration** • **Remove Configuration**

CaliberRDM

This section lists details related to the integration of the CaliberRDM requirements management system.

Item	Description
Status	Status of integration, enabled or disabled.
Hostname	Machine where the external server is installed.
Port	Listening port of the external server.

Item	Description
Username	Credential for the requirements management server.
Password	Credential for the requirements management server.
Project Name	External project with which the Test Manager project is synchronized.
Schedule	Schedule to automatically synchronize requirements with the tool.
Create Requirements	Indicates whether or not the **Enable creation of unassigned requirements** option is active. Enables creation and editing of unmapped requirements in Test Manager projects that are configured for integration with the tool.
Actions	• **Edit Configuration** • **Edit Schedule** • **Edit Notification** • **Disable Configuration** • **Remove Configuration**

IBM® Rational® RequisitePro® Integration

This section lists details related to the integration of the IBM Rational RequisitePro requirements management system.

Item	Description
Status	Status of integration, enabled or disabled.
UNC Project Path	Machine where the external server is installed.
UNC Username	Credential for the UNC file-path, which is required for the integration with IBM Rational RequisitePro.
UNC Password	Credential for the UNC file-path, which is required for the integration with IBM Rational RequisitePro.
User name	Credential for the requirements management server.
Password	Credential for the requirements management server.
Packages	The requirement packages from the external project that are integrated with the Test Manager project.
Requirement Types	Requirement types within the packages that are integrated.
Create Requirements	Indicates whether or not the **Enable creation of unassigned requirements** option is active. Enables creation and editing of unmapped requirements in Test Manager projects that are configured for integration with IBM Rational RequisitePro.
Upload Requirements	Indicates whether or not the **Enable upload of requirements to RequisitePro** option is active. Enables the upload of unmapped/unassigned requirements from Test Manager to RequisitePro. This allows you to upload additional previously unmapped requirement trees to RequisitePro and then have those requirements mapped within Test Manager. When this option is enabled, the **Map Requirement** button in **Requirements** > **Properties** becomes enabled, enabling configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
Property Mappings	Lists any external to internal property mappings that have been defined between the internal and external requirements management systems.
Actions	• **Edit Configuration**

Item	Description
	• **Edit Property Mapping** • **Edit Schedule** • **Edit Notification** • **Disable Configuration** • **Remove Configuration**

SAP Solution Manager

This section lists details related to the integration of the SAP Solution Manager requirements management system.

Item	Description
Status	Status of integration, enabled or disabled.
URL	SAP Solution Manager URL.
Username	Credential for the requirements management server.
Password	Credential for the requirements management server.
Project Name	External project with which the Test Manager project is synchronized.
Schedule	Schedule to automatically synchronize requirements with the tool.
Create Requirements	Indicates whether or not the **Enable creation of unassigned requirements** option is active. Enables creation and editing of unmapped requirements in Test Manager projects that are configured for integration with the tool.
Actions	• **Edit Configuration** • **Edit Schedule** • **Edit Notification** • **Disable Configuration** • **Remove Configuration**

Rally

This section lists details related to the integration of the Rally requirements management system.

Item	Description
Status	Status of integration, enabled or disabled.
URL	URL of Rally.
Username	Credential for the requirements management server.
Password	Credential for the requirements management server.
Project Name	External project with which the Test Manager project is synchronized.
Schedule	Schedule to automatically synchronize requirements with the tool.
Create Requirements	Indicates whether or not the **Enable creation of unassigned requirements** option is active. Enables creation and editing of unmapped requirements in Test Manager projects that are configured for integration with the tool.
Actions	• **Edit Configuration** • **Edit Schedule** • **Edit Notification**

Item	Description
	• **Disable Configuration** • **Remove Configuration**

Telelogic DOORS® Integration

This section lists details related to the integration of the Telelogic DOORS requirements management system.

Item	Description
Status	Status of integration, enabled or disabled.
RM Service URL	The URL of Test Manager's Telelogic DOORS requirement Web Service. The default value should point to the correct location already.
Username	Credential for the requirements management server.
Password	Credential for the requirements management server.
DOORS Installation Path	Client installation path within the front-end server directory structure.
Project Name	External project with which the Test Manager project is synchronized.
Requirement Types	Requirement types within the project that are synchronized.
Schedule	Any defined synchronization schedule.
Create Requirements	Indicates whether or not the **Enable creation of unassigned requirements** option is active. Enables creation and editing of unmapped requirements in Test Manager projects that are configured for integration with Telelogic DOORS.
Upload Requirements	Indicates whether or not the **Enable upload of requirements to Telelogic DOORS** option is active. Enables the upload of unmapped or unassigned requirements from Test Manager to Telelogic DOORS. This allows you to upload additional previously unmapped requirement trees to DOORS and then have those requirements mapped within Test Manager. When this option is enabled, the **Map Requirement** button in **Requirements** > **Properties** becomes enabled, enabling configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
Property Mappings	Lists any external to internal property mappings that have been defined between the internal and external requirements management systems.
Actions	• **Edit Configuration** • **Edit Property Mapping** • **Edit Schedule** • **Edit Notification** • **Disable Configuration** • **Remove Configuration**

Issue Tracking Profiles

Issue tracking profiles enable Test Manager to integrate with external issue tracking systems. The following issue tracking software packages are currently supported by Test Manager “out of the box”:

- SilkCentral Issue Manager
- Borland StarTeam
- IBM Rational® ClearQuest®
- Issue Tracking Web Service. For additional information, refer to the *Test Manager API Help*.

- Bugzilla
- Atlassian JIRA
- Microsoft Team Foundation Server
- Compuware Changepoint

Additional issue tracking systems can be configured by installing a custom plug-in. For additional information, refer to the *SilkCentral Issue Manager API Help*. Defining issue tracking profiles allows you to link tests within the **Tests** area to issues in third-party issue-tracking systems. Linked issue states are updated periodically from the third-party issue tracking system.

Mapping Issue States

After defining a new issue tracking profile, you should map the existing issue states of the external issue tracking system to the issue states of Test Manager. Doing this enables Test Manager to list issues correctly when querying internal and external issues.

To map issue states:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. In the **Actions** column of the issue tracking profile you want to edit, click **Edit mapping of issue tracking profile ...**. The **Edit Status Mapping** dialog box opens.
4. Map internal issue states to corresponding external issue states by selecting the respective entries from the list boxes.

   The existing issue states of the external issue tracking software are listed in the **External** column. The internal issue states of Test Manager are available in the list boxes in the **Internal** column.
5. Once you have mapped each external issue state to an internal state, click **OK** to save your settings, or click **Cancel** to abort the operation.

SilkCentral Issue Manager

This section describes how to configure SilkCentral Issue Manager issue tracking profiles to integrate with Test Manager.

SilkCentral Issue Manager is a robust issue tracking tool that manages bug fixes and enhancement issues related to your company's software projects. Being fully customizable, SilkCentral Issue Manager meets the challenges of your business environment - working across multiple products, releases, and locations. Because SilkCentral Issue Manager's flexible defect tracking workflow allows development and quality assurance to work more closely together, it increases productivity, resulting in an improved development process.

For a list of the SilkCentral Issue Manager versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding SilkCentral Issue Manager Issue Tracking Profiles

To add a SilkCentral Issue Manager issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.

4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Issue Manager** from the **Type** list box, or select **Issue Manager 3.3** to connect to an Issue Manager version 3.3 installation.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Type the **Issue Manager URL** of your Issue Manager installation. This is the URL you use to login to Issue Manager, though without the `login` extension at the end.

   For example, if your standard Issue Manager URL is `http://IssueManager/login`, then the correct service URL is `http://IssueManager`.
9. If you selected **Issue Manager 3.3** from the **Type** list box, proceed with the next step.

   If you selected **Issue Manager** from the **Type** list box, proceed as follows:

   1. Click **Load Projects**. This action will populate the **Project** list box with all initialized Issue Manager projects to which the specified user has access to. Note that only those projects display for which Issue Manager user groups have been defined, and the defined user is a member of at least one user group.
   2. Select the **Project** where Issue Manager issues are maintained.

   Caution: We recommend not to use identical projects for Issue Manager and Test Manager, as this limits flexibility in working with both tools on different future projects.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing SilkCentral Issue Manager Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing Issue Manager issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

These credentials are used to access your issue tracking system.

8. Edit the **Issue Manager URL** of your Issue Manager installation if the location has changed.

   This is the URL you use to login to Issue Manager, though without the `login` extension at the end. For example, if your standard Issue Manager URL is `http://IssueManager/login`, then the correct service URL would be `http://IssueManager`.

9. To change the StarTeam project, click **Load Project** to load all projects from the server and update the **Project** list box, then select a project from the **Project** list box.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

StarTeam Issue Tracking Profiles

This section describes how to configure StarTeam issue tracking profiles to integrate with Test Manager.

StarTeam is a software change management and configuration management tool that enables coordination and management of the software delivery process.

To work with StarTeam profiles and use the go-to-link functionality for change requests in StarTeam, you must have the StarTeam Cross-Platform Client software installed on the computer where the browser is running.

For a list of the StarTeam versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding StarTeam Issue Tracking Profiles

To add a StarTeam issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.

5. Type a **Description** for the new profile.
6. Select **StarTeam** from the **Type** list box
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.

8. Type the **Hostname** of your StarTeam server and the **Port** that is used to connect to the server. If this setting has not been changed, use the default port `49201`.
9. Specify the type of **Encryption** that the profile supports.
10. Click **Load Project** to load all projects from the server and populate the **Project** list box, then select a project from the **Project** list box.

11. Click **Load View** to load all views for the selected project and populate the **View** list box, then select a view from the **View** list box.
12. Click **Load Status Field** to load all enumeration fields for change requests and populate the **Status Field** list box, then select a status field from the **Status Field** list box.

    If you are using a custom workflow in StarTeam, this field is the workflow driver field in StarTeam that maps to the Test Manager issue state.
13. Select a type of link from the **Link Type** list box.

starteam://	**External ID** links on the **Issues** tab will open the cross platform client.
http://	**External ID** links on the **Issues** tab will open the issue in the StarTeam web UI.
14. If you selected `http://` in the **Link Type** list box, enter the web server address of the StarTeam web UI in the **WebServer** field.
15. Select `Yes` or `No` in the **Workflow** field. Select `Yes` to show the required fields that are specified in the workflow of the selected **View**.
16. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
17. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing StarTeam Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing StarTeam issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the **Hostname** of your StarTeam server and the **Port** that is used to connect to the server.
9. Modify the type of **Encryption** that the profile supports.
10. To change the StarTeam project, click **Load Project** to load all projects from the server and update the **Project** list box, then select a project from the **Project** list box.
11. To change the view, click **Load View** to load all views for the selected project and populate the **View** list box, then select a view from the **View** list box.

12. To change the workflow driver field, click **Load Status Field** to load all enumeration fields for change requests and populate the **Status Field** list box, then select a status field from the **Status Field** list box.

    If you are using a custom workflow in StarTeam, this field is the workflow driver field in StarTeam that maps to the Test Manager issue state.

13. Select a type of link from the **Link Type** list box.

starteam://	**External ID** links on the **Issues** tab will open the cross platform client.
http://	**External ID** links on the **Issues** tab will open the issue in the StarTeam web UI.

14. If you selected `http://` in the **Link Type** list box, enter the web server address of the StarTeam web UI in the **WebServer** field.
15. Select `Yes` or `No` in the **Workflow** field. Select `Yes` to show the required fields that are specified in the workflow of the selected **View**.
16. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

17. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Atlassian JIRA

This section describes how to configure Atlassian JIRA (JIRA) issue tracking profiles to integrate with Test Manager.

The JIRA plug-in relies on the SilkCentral Java API for integration. Test Manager communicates with JIRA through the JIRA Web-service by using the Apache Axis library.

Note: See the sources of the package `com.borland.sctm.issuetracking.jira` to see how these elements fit together.

For a list of the JIRA versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding Atlassian JIRA Issue Tracking Profiles

To integrate Test Manager with JIRA, enable the System Soap Services plug-in in JIRA. For detailed information, refer to the JIRA documentation.

To add a JIRA issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.

5. Type a **Description** for the new profile.
6. Select **Atlassian JIRA** from the **Type** list box.

7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. In the **Endpoint** field, type the hostname of your JIRA server and the port that is used to connect to the server.
9. Click **Load Project** to load all projects from the server and populate the **Project** list box, then select a project from the **Project** list box.
10. Optional: Select **true** from the **Show custom fields** list to display custom JIRA fields in the issue dialog. If true is selected, the JIRA user defined for the issue tracking profile has to have JIRA administration permissions.
11. Optional: If you experience performance issues on large JIRA installations, select **false** from the **Update Daily Issue Statistics** list. The daily issue statistics will not be updated.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
13. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing Atlassian JIRA Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing Atlassian JIRA issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. In the **Endpoint** field, edit the hostname of your JIRA server and the port that is used to connect to the server.
9. To change the JIRA project, click **Load Project** to load all projects from the server and update the **Project** list box, then select a project from the **Project** list box.
10. Optional: Select **true** from the **Show custom fields** list to display custom JIRA fields in the issue dialog. If true is selected, the JIRA user defined for the issue tracking profile has to have JIRA administration permissions.
11. Optional: If you experience performance issues on large JIRA installations, select **false** from the **Update Daily Issue Statistics** list. The daily issue statistics will not be updated.
12. Click **OK**.

Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

13. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Bugzilla

This section describes how to configure Bugzilla issue tracking profiles to integrate with Test Manager.

The Bugzilla plug-in relies on the SilkCentral Java API for integration. Test Manager communicates with Bugzilla through the XML-RPC Bugzilla Web-service introduced with Bugzilla 3.0 by using the Redstone XML-RPC library. To enable Bugzilla integration, ensure that you have installed a SOAP::Lite package for the web server. For example, for the Debian distribution install the `libsoap-lite-perl` package.

For a list of the Bugzilla versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding Bugzilla Issue Tracking Profiles

To add a Bugzilla issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Bugzilla** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Enter the URL of your Bugzilla installation. For example, `http://bugzillaserver/cgi-bin/bugzilla3`.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
10. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing Bugzilla Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing Bugzilla issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the URL of your Bugzilla installation.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
10. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

IBM Rational ClearQuest

This section describes how to configure IBM Rational ClearQuest issue tracking profiles to integrate with Test Manager.

IBM Rational ClearQuest products provide flexible defect/change tracking and automated workflow support. The two key products are IBM Rational ClearQuest (ClearQuest) and IBM Rational ClearQuest MultiSite (ClearQuest Multisite). To work with ClearQuest profiles, you must have the ClearQuest client software installed on the computer where the SilkCentral front-end server is running. For detailed information about installing ClearQuest, refer to the ClearQuest documentation.

For a list of the ClearQuest versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding ClearQuest Issue Tracking Profiles

To add a ClearQuest issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.

3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **IBM Rational ClearQuest** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Enter the **Repository Info** of your ClearQuest installation.

   This is the database name that is defined in the ClearQuest client software.
9. Specify the **Record Type**, which is the issue type of ClearQuest.

   When entering an issue in Test Manager, ClearQuest will save the issue with the issue type you define in this text box. The default issue type is `Defect`.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing ClearQuest Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing ClearQuest issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the **Repository Info** of your ClearQuest installation.

   This is the database name that is defined in the ClearQuest client software.
9. Change the **Record Type**, which is the issue type of ClearQuest.

   When entering an issue in Test Manager, ClearQuest saves the issue with the issue type you define in this field.
10. Click **OK**.

Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Team Foundation Server

This section describes how to configure Team Foundation Server (TFS) issue tracking profiles to integrate with Test Manager.

For a list of the TFS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Installing a Team Foundation Server Web Service Proxy

To communicate with a TFS, you need to install a Team Foundation Server Web Service Proxy as an interface to the TFS.

To install a TFS proxy service:

1. Install Microsoft Visual Studio Team Explorer 2010 from *http://www.microsoft.com/downloads/details.aspx?FamilyID=fe4f9904-0480-4c9d-a264-02fedd78ab38*.
2. Install IIS and the Web Deployment Tool, for example with the Web Platform Installer from *http://www.microsoft.com/web/downloads/default.aspx*.
3. Download the Team Foundation Server Web Service Proxy from **Help** > **Tools**.
4. Unzip the downloaded package.
5. Open a command shell and type `DotNetTfsWebServiceProxy.deploy.cmd /Y` to install the proxy service. If an error message box displays, stating that ASP.NET 4 is required, refer to *http://msdn.microsoft.com/en-us/library/k6h9cz8h.aspx*.
6. Open the IIS Manager.
7. Verify that the new website exists.
8. In the root folder of the virtual directory, open the `Web.config` file.
9. In the *appSettings* section of the file, modify the value of the key *WorkItemTrackingCacheRoot* to a local directory.

   For example:

   ```
   <appSettings>
     <add key="WorkItemTrackingCacheRoot" value="C:\temp" />
   </appSettings>
   ```

10. If the directory you have specified does not exist, create it.
11. Grant **Everyone** full rights on the specified directory.

For additional information on deploying an ASP.NET Web Application, refer to *http://msdn.microsoft.com/en-us/library/ee942158.aspx*.

Adding TFS Issue Tracking Profiles

You need to configure a Team Foundation Server Web Service Proxy, which is provided by Micro Focus to enable Test Manager to access TFS issues.

Important: This is not the Team Foundation Server Proxy provided by Microsoft.

To add a TFS issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Team Foundation Server 2010** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Type the **Domain** of the TFS user.
9. Enter the URL of your TFS installation.

   For example, `http://tfsserver:8080/tfs`.
10. Type the **Collection** to which your project belongs.

    For example `DefaultCollection`.

    All projects in the collection are listed in the **Projects** list box.
11. Select the **Project** from the list box.
12. Type the URL of your TFS proxy. For example `http://tfsproxyserver/DotNetTfsWebServiceProxy_deploy/TfsWebServiceProxy.asmx`.
13. Click **Load Work Item Type**. The **Work Item Type** list box is populated with the available work item types.
14. Select the **Work Item Type** from the list box.
15. Click **Load Initial State**. The **Initial State** list box is populated with the states that are allowed for the selected work item type.
16. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
17. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing TFS Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing TFS issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.

3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the **Domain** of the TFS user.
9. Edit the **URL** and **Collection** of your TFS installation.
10. Select a different **Project**.
11. Edit the URL and port of your TFS proxy.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

Changepoint

This section describes how to configure Changepoint issue tracking profiles.

Changepoint is an IT management and governance tool that enables organizations to maximize the business value of the entire IT portfolio including projects, applications, and infrastructure.

This integration allows you to connect Test Manager to Changepoint to store issues.

Important: The Changepoint environment must have the Changepoint API installed and the `CPWebService` configured.

Known Issues

Workflow Do not change the Changepoint workflows for requests after submitting them in Test Manager. Do not use different workflows in requests when assigning external issues. The filter criteria used in workflows should be based on fields used in the Test Manager **Issue Tracking Profile**. Otherwise, unexpected states may be assigned to requests in Test Manager.

Synchronizing the Time for the Changepoint and Test Manager Servers

This section describes how to ensure that the Changepoint server and Test Manager server communicate properly.

1. Open the `web.config` file located in the `../Changepoint/CP Web Services` directory.
2. Add the following key to the `security` section:

   ```
   <timeToleranceInSeconds>86400</timeToleranceInSeconds>
   ```
3. Save the file.

Opening Changepoint Issues in Context

In order to open Test Manager issues in the Changepoint system, you need to follow the steps below to install a connection file on the Changepoint server.

1. In Test Manager, click **SilkCentral** > **Help** > **Tools** to view the **Downloadable Client Tools** page.

2. Click the `Changepoint Request Form` link.
3. When prompted, choose to save the `ChangepointRequestForm.zip` file to disk.
4. Extract `CPRequestForm.html` from `ChangepointRequestForm.zip`.

   If you are using Changepoint 2010, the required file is in the `Changepoint 2010` folder. It is called `CPRequestForm2010.html`. Extract this file and manually rename it to `CPRequestForm.html`.
5. Manually copy `CPRequestForm.html` to the Changepoint server and placed into the virtual directory in which Changepoint is running.

Creating a Changepoint Issue Tracking Profile

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Compuware Changepoint** from the **Type** list box.
7. Type a valid **Username** and **Password** for the issue tracking system.

   Note: In order to select a project and to submit a new issue, the **Username** field must contain a Changepoint user who has the project's `Edit Project Plan` permission.
8. Enter the URL of your Changepoint installation in the **Changepoint Server URL** field.
9. Enter the URL of your Changepoint web service server in the **Changepoint WebService URL** field.
10. Click **Load Initiator** to load your Changepoint initiators.
11. Select an initiator from the **Initiator** list box.
12. Click **Load Client** to load your Changepoint clients.
13. Select a client from the **Client** list box.
14. Click **Load Initiative** to load your Changepoint initiatives.
15. Select an initiative from the **Initiative** list box.
16. Click **Load Project** to load your Changepoint projects.
17. Select a project from the **Project** list box.
18. Click **Load Request Type** to load your Changepoint request types.
19. Select a request type from the **Request Type** list box.
20. Click **Load Application** to load your Changepoint applications. Select an application from the **Application** list box.
21. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
22. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Deleting Issue Tracking Profiles

To delete an issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. In the **Actions** column of the issue tracking profile you wish to delete, click **Delete issue tracking profile ...**. A confirmation dialog box displays.
4. Click **Yes**.

Issue Tracking Profiles Page

Project:<Project Name> > **Project Settings** > **Issue Tracking**

Use this page to configure profiles for the integration of external issue tracking systems into Test Manager. Click **New Profile** to create a new issue tracking profile. For each issue tracking profile, the page displays the following columns:

Column	Description
Name	The name of the profile as it displays in the SilkCentral GUI and in reports. Click the name to edit a profile.
Type	The external issue tracking system.
Login	The login name with which SilkCentral connects to the issue tracking system.
Repository Info	The physical location of the issue tracking system. Hostname or URL.
Created On	Date when the issue tracking profile was created.
Created By	The user who created the issue tracking profile.
Actions	The following actions can be performed on an issue tracking profile: • **Delete issue tracking profile** • **Edit mapping of issue tracking profile**

Source Control Profiles

Source control profiles enable Test Manager to integrate with external source control systems. Defining source control profiles allows you to define where Test Manager's execution servers should retrieve program sources for test execution.

You need double the amount of free disk space on the execution server to accommodate the source files because SilkCentral checks out the source control tree and then generates a working directory with the executable source files.

The following source control systems are currently supported by Test Manager "out of the box":

- Borland StarTeam
- SilkTest Workbench
- Serena® Version Manager™ (PVCS®)
- Concurrent Version System (CVS)
- Microsoft® Visual SourceSafe® (MSVSS)
- Universal Naming Convention (UNC) (file-system access)

- Subversion
- Apache Commons Virtual File System (VFS)
- TestPartner

Additional source control systems can be configured by installing a custom plug-in. For detailed information, refer to the *Test Manager API Help*.

TestPartner

This section describes how to configure TestPartner source control profiles.

TestPartner is a Micro Focus product that tests Web- and Microsoft Windows-based applications that use Microsoft technologies. TestPartner records user actions to quickly produce powerful tests. Each recorded test displays as a series of actions in clear, concise steps that can be easily understood by all testers, from novice to expert. You can record user sessions with the application, add validation functions, and replay the sessions later to ensure that the application works as expected.

Adding TestPartner Source Control Profiles

To create a TestPartner source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **TestPartner** from the **Source control system** list box.

   Note: When connecting to a TestPartner Oracle database, there is a TestPartner requirement that the TNS name must be the same as the host server name.
6. Select a database type from the **Database type** list box, either SQL Server or Oracle.
7. Type the name of the database server in the **Database server** text box.
8. Type the name of the database in the **Database name** text box.
9. Type the port number of the database server in the **Database port** text box.
10. Type the database schema or owner name in the **Database schema/owner name** text box.
11. Type the name of the database user in the **Database user** text box.
12. Type the password for the database user in the **Database password** text box.
13. Type a valid **Username** and **Password**.

    These credentials will be used to access your repository.
14. Type the **Working folder** to which the Test Manager execution server should copy the source files.

    The working folder must be a local path. For example, `C:\TempSources\`.
15. Use the **Project path** text box to filter which scripts from the database are available to use as tests. Click **Browse** to display the **Select Project Path** dialog box.

    The **Select Project Path** dialog box provides a tree of three levels that you can choose from:

First level	The entire TestPartner database. Note that if you select this level, the **Project path** text box will remain blank.
Second level	The TestPartner project.
Third level	The script type within a specified TestPartner project.

Select one of the preceeding options and click **OK**. The path will be added to the **Project Path** text box.

16. Click **OK**.

Editing TestPartner Source Control Profiles

To modify a TestPartner source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Choose from the following options:

   - Edit the TestPartner **Database type**.
   - Edit the TestPartner **Database server**.
   - Edit the TestPartner **Database name**.
   - Edit the TestPartner number of the TestPartner database server.
   - Edit the TestPartner **Database schema/owner name**.
   - Edit the TestPartner **Database user**.
   - Edit the TestPartner **Database password**.
6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
7. Edit the **Project path** you want this profile to use.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Click **OK**.

Serena Version Manager (PVCS)

This section describes how to configure Serena Version Manager (PVCS) source control profiles.

Serena Version Manager, from the makers of PVCS, is the full-featured solution for version control and revision management in software projects. More than simply storing code revisions, Version Manager is a robust, full-featured solution with security, high performance, and varying levels of support for distributed teams. For information on the supported versions, refer to the *Test Manager Release Notes*.

Adding PVCS Source Control Profiles

To create a PVCS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.

5. Select **PVCS** from the **Source control system** list box.
6. Type the UNC path of the PVCS **Repository** you want to access.

   If you do not know the UNC path of the repository, consult your PVCS administrator.
7. Type a valid **UNC username** and **UNC password**.

   These credentials are required to access the UNC path of the configuration file.
8. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Type the **Execution path**. This is the local path of the PVCS installation, where the command line tool `pcli.exe` is located. The default path is `C:\Program Files\Serena\vm\win32\bin`.

   Note: The PVCS client software must be installed on the front-end server and each execution server. PVCS must be installed in identical paths on each machine. For example, if you install PVCS on the TestPartner front-end server to `C:\Program Files\Serena\`, you must install PVCS in the same path on the execution servers.
10. Type a valid **Username** and **Password**.

    These credentials will be used to access your repository.
11. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing PVCS Source Control Profiles

To modify a PVCS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the UNC path of the PVCS **Repository**. If you do not know the UNC path of the repository, consult your PVCS administrator.
6. Edit the **UNC username** and **UNC password** as required. These credentials are required to access the repository UNC path you specified above.
7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Edit the **Execution path**. This is the local path of the PVCS installation, where the command line tool `pcli.exe` is located. The default path is `C:\Program Files\Merant\vm\win32\bin`.

   Note: The PVCS client software must be installed on the front-end server and each execution server. PVCS must be installed in identical paths on each machine. For example, if you install

PVCS on the Test Manager front-end server to `C:\Program Files\Serena\`, you must install PVCS in the same path on the execution servers.

9. Edit the **Username** and **Password**.

   These credentials are used to access your repository.

10. Edit the **Project path** you want this profile to use.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

StarTeam Source Control Profiles

This section describes how to configure StarTeam source control profiles.

StarTeam promotes team communication and collaboration through centralized control of all project assets. Protected yet flexible access ensures that team members can work whenever and wherever they like through an extensive choice of Web, desktop, IDE, and command-line clients. StarTeam offers a uniquely comprehensive solution that includes integrated requirements management, change management, defect tracking, file versioning, threaded discussions, and project and task management.

For a list of the StarTeam versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding StarTeam Source Control Profiles

To create a StarTeam source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.

5. Select **StarTeam** from the **Source control system** list box.
6. Type the **Hostname** of your StarTeam server.
7. Type the **Port** that is to be used to connect to the StarTeam server.

   If the port is not changed, use the default port `49201`.

8. Type a valid **Username** and **Password**.

   These credentials will be used to access your repository.

9. Specify if the profile supports **Encryption**.
10. Type the **Working folder** to which the Test Manager execution server should copy the source files.

    The working folder must be a local path. For example, `C:\TempSources\`.

11. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
12. Click **OK**.

Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing StarTeam Source Control Profiles

To modify a StarTeam source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Choose from the following options:

   - Edit the **Hostname** of your StarTeam server.
   - Edit the port that is to be used to connect to the StarTeam server. If the port is not changed, use the default port `49201`.
   - Specify if the profile supports **Encryption**.
6. Edit the **Project path** you want this profile to use.
7. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Concurrent Version System (CVS)

This section describes how to configure CVS source control profiles.

CVS is a powerful source control tool that handles complete source code trees. It can be customized using scripting languages such as PERL and Korn. CVS is decentralized so that users can maintain their own source directory trees. It also enables concurrent file editing.

For a list of the CVS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding CVS Source Control Profiles

To create a CVS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Choose **CVS** from the **Source control system** list box.
6. Type the CVS server name or IP address in the **Hostname** text box.
7. Type the port that is to be connected to in the **Port** text box.
8. Specify the connection method in the **Method** text box.

   Currently, the `ext`, `pserver`, and `local` connection methods are supported.

   This makes the **Port** setting optional.
9. Specify the URL of the CVS **Repository** you want to access.

   For example, `/var/lib/cvs`. If you do not know the URL of the repository, please consult your CVS administrator.
10. Type a valid CVS **Username** and **Password**.

    These credentials will be used to access your CVS repository.

    Note: These settings are optional when using the `ext` connection method.
11. Specify the CVS **Module** that is to be used.
12. Type the **Working folder** to which the Test Manager execution server should copy the source files.

    The working folder must be a local path. For example, `C:\TempSources\`.
13. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
14. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing CVS Source Control Profiles

To modify a CVS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Choose from the following options:

   - Edit the CVS server name or IP address in the **Hostname** text box.

- Edit the port that is to be connected to in the **Port** text box.
- Edit the URL of the CVS **Repository** you want to access. If you do not know the URL of the repository, consult your CVS administrator.
- Edit the CVS **Module** that is to be used.

6. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.

   Note: These settings are optional when using the `ext` connection method.

7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.

8. Specify the connection method in the **Method** text box.

   Currently, the `ext`, `pserver`, and `local` connection methods are supported.

   This makes the **Port** setting optional.

9. Edit the **Project path** you want this profile to use.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Microsoft Visual SourceSafe

This section describes how to configure Microsoft Visual SourceSafe (MSVSS) source control profiles.

MSVSS is a version-control system for managing software and Web-site development. Fully integrated with the Visual Basic-, Visual C++-, Visual J++-, Visual InterDev-, and Visual FoxPro development environments, as well as with Microsoft Office applications, MSVSS provides easy-to-use, project-oriented version control. MSVSS works with any file produced with any development language, authoring tool, or application. Users can work at both the file and project level while promoting file reuse.

For a list of the MSVSS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding MSVSS Source Control Profiles

Tip: SourceSafe clients must be installed on all front-end, application, and execution servers.

To create a MSVSS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.

5. Select **MSVSS** or **MSVSS (cmd line)** from the **Source control system** list box.

`MSVSS (cmd line)` utilizes the MSVSS command line plug-in, which works exactly like MSVSS, except that SilkCentral users are automatically logged out of MSVSS when the user logs out from SilkCentral. When selecting `MSVSS`, SilkCentral users remain logged in to MSVSS for an indefinite time.

6. If you selected **MSVSS (cmd line)**, specify the location of the **SourceSafe executable** `ss.exe`.

   SourceSafe must be installed identically on all execution servers and the front-end server. This allows you to specify a definite path. For example, `C:\Program Files\Microsoft Visual Studio\VSS\win32\ss.exe`. If SourceSafe is installed in different locations, see *Configuring the location of a SourceSafe Client*.
7. In the **SourceSafe database (srcsafe.ini)** text box, type the UNC path and file name of the SourceSafe configuration file you want to access or click **Browse** to locate the SourceSafe configuration file..

   Note: SourceSafe configuration files use the name `srcsafe.ini`.
8. Type a valid **UNC username** and **UNC password**.

   These credentials are required to access the UNC path of the configuration file.
9. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
10. Type a valid **Username** and **Password**.

    These credentials will be used to access your repository.
11. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing MSVSS Source Control Profiles

To modify a MSVSS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. If you selected **MSVSS (cmd line)**, specify the location of the **SourceSafe executable** `ss.exe`.

   SourceSafe must be installed identically on all execution servers and the front-end server. This allows you to specify a definite path. For example, `C:\Program Files\Microsoft Visual Studio\VSS\win32\ss.exe`. If SourceSafe is installed in different locations, see *Configuring the location of a SourceSafe Client*.
6. In the **SourceSafe database (srcsafe.ini)** text box, edit the UNC path and file name of the SourceSafe configuration file, or click **Browse** to locate the file.

   If you do not know the location of the configuration file, consult your SourceSafe administrator.

Note: SourceSafe configuration files use the name `srcsafe.ini`.

7. Edit the **UNC username** and **UNC password**.
   These credentials are required to access your configuration file's UNC path.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.
   The working folder must be a local path. For example, `C:\TempSources\`.
9. Edit the **Username** and **Password**.
   These credentials are used to access your repository.
10. Edit the **Project path** you want this profile to use.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Configuring the Location of a SourceSafe Client

To configure the location of a SourceSafe client:

1. In the **SourceSafe executable** text box, type `ss.exe` without any path information.
2. On each execution server and on the front-end server, type the local path of the SourceSafe executable `ss.exe` to the Windows system path. To do this, click **Start** > **Settings** > **Control Panel** > **System** . The **System Properties** dialog box displays.
3. Click the **Advanced** tab and click **Environment Variables**. The **Environment Variables** dialog box displays.
4. Select the **Path** variable in the **System variables** section and click **Edit**.
5. Add the local path of the SourceSafe executable to the list of existing **Variable values**. You can append a new variable value to existing values by entering a semicolon (`;`) followed by the path information.
6. Repeat this procedure for each execution server and for the front-end server.

Subversion

This section describes how to configure Subversion (SVN) source control profiles.

Subversion (SVN) is the successor to the Concurrent Versions System (CVS). Subversion manages versions using transaction numbers. With each commit, the transaction number is incrementally increased.

What other source control systems call *labels*, Subversion refers to as *tags*. These tags are encoded in the Subversion URL. For example, `http://MyHost/svn/MyApp/trunk` is a Subversion URL and `http://MyHost/svn/MyApp/tags/build1012` is a Subversion tag.

Test Manager supports Subversion tags. If a Subversion URL contains the `trunk` directory, you can define a label `tags/build1012`. This label replaces `trunk` in the Subversion URL.

Note: If a Subversion URL does not contain `trunk` and you define a label, Test Manager throws an error.

Adding Subversion Source Control Profiles

To create a Subversion source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Choose **Subversion** from the **Source control system** list box.
6. Type the URL of the Subversion **Repository** you want to access.

   If you do not know the URL of the repository, consult your Subversion administrator.

   Note: To use SSH, add *ssh* to the URL, for example `svn+ssh://<hostname>:<port>`.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access your repository.
8. Type a valid **SSH username** and **SSH password** or **SSH keyfile**.

   These credentials are used to access the SSH server. The password overrides the keyfile, so if you only have a keyfile, leave the **SSH password** text box empty. If you use a keyfile, the path to the keyfile must be valid on every execution server that uses the source control profile.
9. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
10. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing Subversion Source Control Profiles

To modify a Subversion source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **URL** of the Subversion repository you want to access.

   If you do not know the URL of the repository, please consult your Subversion administrator.

   Note: If you cannot check out files, when editing the URL of the Subversion **Repository**, delete the source control mirrors directory on your execution server. For example, `C:\Documents and Settings\All Users\Application Data\Silk\SCC35\SrcCtrlMirrors`.

Note: To use SSH, add *ssh* to the URL, for example `svn+ssh://<hostname>:<port>`.

6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
7. Type a valid **SSH username** and **SSH password** or **SSH keyfile**.

   These credentials are used to access the SSH server. The password overrides the keyfile, so if you only have a keyfile, leave the **SSH password** text box empty. If you use a keyfile, the path to the keyfile must be valid on every execution server that uses the source control profile.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Edit the **Project path** you want this profile to use.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Universal Naming Convention

This section describes how to configure Universal Naming Convention (UNC) source control profiles.

Short for Universal Naming Convention or Uniform Naming Convention, UNC is a PC format for specifying the location of resources on a local-area network (LAN). UNC uses the following format: `\\server-name\shared-resource-pathname`.

For example, to access the file `test.txt` in the directory `examples` on the shared server `silo`, you would write: `\\silo\examples\test.txt`.

You can also use UNC to identify shared peripheral devices, such as printers. The idea behind UNC is to provide a format so that each shared resource can be identified with a unique address.

UNC is only supported on Microsoft Windows operating systems. If you plan to use a non-windows execution server you can use the Apache Commons VFS source control profile instead.

Adding UNC Source Control Profiles

To create a UNC source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **UNC** from the **Source control system** list box.
6. Type the **UNC path** that you want to access.

   This is the path to the location where your test sources are located.
7. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Type a valid **UNC username** and **UNC password**.

These credentials are required to access the UNC path of the configuration file.

9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing UNC Source Control Profiles

To modify a UNC source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.

5. Edit the **UNC path**.

   This is the path to where your test sources are located.

6. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.

7. Edit the **UNC username** and **UNC password**.

   These credentials are required to access your configuration file's UNC path.

8. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Apache Commons Virtual File System

This section describes how to configure Virtual File System (VFS) source control profiles.

A VFS is an abstraction layer on top of a more concrete file system. The purpose of a VFS is to allow client applications to access different types of concrete file systems in a uniform way. Apache Commons VFS provides a single API for accessing various file systems. It presents a uniform view of the files from various sources. The protocols that are currently supported for VFS by Test Manager are:

Protocol	Description
http	Copies the given file. This protocol type is also supported for copying and unpacking ZIP, JAR, or other zipped files. It is required to specify a .zip file on a http server. For example, `zip:http://myTestServer/myTests.zip`. The .zip file will be extracted on the execution server.

Protocol	Description
ftp	Copies the given file. This protocol type is also supported for copying and unpacking ZIP, JAR, or other zipped files.
smb	Server Message Block (smb) copies all files and folders. This protocol can be used instead of a UNC profile. For example, the VFS smb path `smb://server-name/shared-resource-path` is equivalent to the UNC path `\\server-name\shared-resource-path`.

Note: When you create a new ProcessExecutor test that uses VFS for source control, you need to specify the complete path to the executable in the **Executable Name** text box.

Adding VFS Source Control Profiles

To create a VFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **VFS** from the **Source control system** list box.
6. Type the URL of the VFS **Repository** you want to access.

   Specify the appropriate protocol type in the URL:

FTP	`ftp://<ftp server URL>`
HTTP	http://<http server URL>
SMB	smb://<Samba server url>

   Note: This field is case sensitive.

   Note: HTTP, FTP and SMB are also supported for zipped files. In order to point to a zipped file the URL must be adjusted to `<zipped file type>:<protocol>://<server URL pointing to zipped file>` to include the type of the zipped file. For example, `zip:http://193.80.200.135/<path>/archive.zip` or `jar:http://193.80.200.135/<path>/archive.jar`.
7. Type a valid VFS **Username** and **Password**.

   These credentials will be used to access your VFS repository. The SMB protocol allows including the domain name in the username in the following form: `domain/username`.
8. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing VFS Source Control Profiles

To modify a VFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the URL of the VFS **Repository** you want to access.

   Note: This field is case sensitive.
6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Edit the **Project path** you want this profile to use.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

SilkTest Workbench

This section describes how to configure SilkTest Workbench source control profiles.

SilkTest Workbench is an automated testing tool that accelerates the functional testing of complex applications. SilkTest Workbench provides support for testing applications developed in a wide variety of development tools including Java, .NET, browser-based web applications, and COM components, including both ActiveX controls and automation objects. With SilkTest Workbench, you can record user sessions with your applications to create tests, enhance the test by adding validation and test logic, and play back tests to ensure that the applications work as expected.

Adding SilkTest Workbench Source Control Profiles

To create a SilkTest Workbench source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.

3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **SilkTest Workbench Test** from the **Source control system** list box.
6. Select a database type from the **Database type** list box, either SQL Server or Oracle.

   Note: The Microsoft Office Access database type is not supported. When connecting to a SilkTest Workbench Oracle database, there is a SilkTest Workbench requirement that the TNS name must be the same as the host server name.
7. Enter the name of the SilkTest Workbench database server in the **Database server** text box.
8. Enter the name of the SilkTest Workbench database in the **Database name** text box.
9. Enter the port number of the SilkTest Workbench database server in the **Database port** text box.
10. Enter the SilkTest Workbench database schema or owner name in the **Database schema/owner name** text box.
11. Enter the name of the database user in the **Database user** text box.
12. Enter the password for the database user in the **Database password** text box.
13. Enter the name of a valid SilkTest Workbench user in the **SilkTest Workbench User name** text box.
14. Enter the password for the SilkTest Workbench user in the **SilkTest Workbench password** text box.
15. Type the **Working folder** to which the Test Manager execution server should copy the source files.

    The working folder must be a local path. For example, `C:\TempSources\`.
16. Click **Retrieve**. All projects are listed in the **Projects** list.
17. Select one or more projects.
18. Click **OK**.

Editing SilkTest Workbench Source Control Profiles

To modify a SilkTest Workbench source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the SilkTest Workbench **Database type**.

   Note: The Microsoft Office Access database type is not supported.
6. Choose from the following options:

   - Edit the SilkTest Workbench **Database server**.
   - Edit the SilkTest Workbench **Database name**.
   - Edit the **Database port** number of the SilkTest Workbench database server.
   - Edit the SilkTest Workbench **Database schema/owner name**.
   - Edit the SilkTest Workbench **Database user**.
   - Edit the SilkTest Workbench **Database password**.
   - Edit the SilkTest Workbench **User name**.
   - Edit the SilkTest Workbench **Password**.

7. Edit the **Working folder** to which the Test Manager execution server copies the source files.
   The working folder must be a local path. For example, `C:\TempSources\`.
8. Select one or more projects from the **Projects** list.
9. Click **OK**.

Microsoft Team Foundation Server 2010

This section describes how to configure Team Foundation Server (TFS) source control profiles.

For a list of the TFS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding TFS Source Control Profiles

To add a TFS source control profile, the Microsoft Visual Studio Team Explorer Everywhere 2010 command-line client needs to be installed on the front-end server and every execution server, on which you want to use the source control profile. You can download the CLC from the *Microsoft Download Center*.

To create a TFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.
   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **Team Foundation Server 2010** from the **Source control system** list box.
6. Enter the **URL** of your TFS installation.
   For example, `http://tfsserver:8080/tfs`.
7. Type the **Domain** of the TFS user.
8. Type a valid **Username** and **Password**.
   These credentials will be used to access your repository.
9. Type the **Working folder** to which the Test Manager execution server should copy the source files.
   The working folder must be a local path. For example, `C:\TempSources\`.
10. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing TFS Source Control Profiles

To add a TFS source control profile, the Microsoft Visual Studio Team Explorer Everywhere 2010 command-line client needs to be installed on the front-end server and every execution server, on which

you want to use the source control profile. You can download the CLC from the *Microsoft Download Center*.

To modify a TFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the URL of the TFS **Repository** you want to access.
6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Edit the **Project path** you want this profile to use.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Deleting Source Control Profiles

To remove a source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. In the **Actions** column of the issue tracking profile you wish to delete, click **Delete source control profile ...**. A confirmation dialog box displays.
4. Click **Yes**. You are returned to the **Source Control** page.

Source Control Profiles Page

Project:<Project Name> > **Project Settings** > **Source Control**

Use this page to configure profiles to integrate external source control systems with Test Manager. Click **New Profile** to create a new source control profile. For each source control profile, the page displays the following columns:

Item	Description
Name	The name of the profile as it displays in the SilkCentral GUI and in reports. Click the name to edit a profile.

Item	Description
Type	The external source control system.
Working Folder	Local or mapped working folder to which temporary sources are checked out to.
Created On	Date when the source control profile was created.
Created By	The user who created the source control profile.
Changed On	Date when the source control profile was last modified.
Changed By	The user who last modified the source control profile.
Actions	**Delete source control profile**

Data Sources for Data-Driven Tests

Data-driven tests are tests that are derived from values in an existing data source, such as a spreadsheet or a database. The data sources are managed in a project-specific scope.

Tip: To aknowledge changes in your data source to Test Manager, you must synchronize your data source profile with your data source whenever your data source is updated or changed.

Configuring JDBC Data Sources

Caution: The concatenated length of all primary keys may not exceed 255 characters in length for JDBC data sources.

To configure a JDBC data source:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Data Sources** tab. The **Data Sources** page displays, listing all of the data sources that have been created for the system.
3. Click **New Data Source** to open the **New Data Source** dialog box.
4. Type a **Name** for the data source.
5. From the **Data source type** list box, select `JDBC`.

   Note: If you are setting up an ODBC data source, you need to manually insert your ODBC **Driver class** and **URL**. For example, **Driver class**: `sun.jdbc.odbc.JdbcOdbcDriver`, **URL**: `jdbc:odbc:MyDatabaseName`. You must also set up an ODBC data source in MS Windows in the **Administrative Tools**. For more information, refer to Microsoft Windows Help. If you have your front-end server and your application server on different machines, make sure that the name of your system data source set-up in Microsoft Windows is the same as the ODBC data source. These names are case-sensitive.
6. If you select **JDBC** as the **Data source type**, the **Driver class** text box is populated automatically. In the **URL** text box, replace the host name value, `<hostname>`, with the name of the computer that is hosting the data source and replace the database name value, `<databasename>`, with the name of the target database.
7. In the **Username** and **Password** text boxes, enter valid database credentials.
8. *Optional:* If you are working with a database that includes multiple tables, and you want to narrow down the data source to specific tables, you can browse to and select specific tables for inclusion:
   1. Click **[...]** next to the **Table filter** text box.
   2. The **Select Table Filter** dialog box displays. Select the tables that you want included as your data source.

3. Click **OK**.

9. *Optional:* **Key column** selection is used by tests to define which worksheet columns within a data source are used as primary key. This is helpful if your data source will undergo edits, for example when you add or remove rows within a worksheet. Even if your data source is edited, tests will still be able to identify which columns or rows should be used. Tests created from data-driven data sources use key column values in their names, rather than column numbers.

   To configure a key column:

   1. Click **[...]** next to the **Key column** text box.
   2. The **Select Key Column** dialog box displays. Select a column from the column list that is to act as a key column.
   3. Click **OK**.

10. Click **OK** on the **New Data Source** dialog box.

Configuring Microsoft Excel or CSV Data Sources

Caution: Excel worksheets using password protection can not be configured as data sources for Test Manager. Turn off a worksheet's password protection to use it as data source for data-driven testing. Values within any cell of an Excel or CSV data source may not exceed 255 characters in length. Additionally, the concatenated length of all primary keys may also not exceed 255 characters in length for Excel and CSV.

To configure a Microsoft Excel or CSV data source:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Data Sources** tab. The **Data Sources** page displays, listing all of the data sources that have been created for the system.
3. Click **New Data Source** to open the **New Data Source** dialog box.
4. Type a **Name** for the data source.
5. From the **Data source type** list box, select **MS Excel** to configure a Microsoft Excel data source, or select **CSV** to configure a CSV data source.
6. From the **Source control profile** list box, select the pre-configured source control profile that hosts your data file. For detailed information regarding the configuration of source control profiles, see *Source Control Profiles*.
7. Click **Browse** to open the **Select Source Control Path** dialog box. Browse to and select a data source file of the selected type in your source control path.
8. *Optional:* MS Excel only. If you are working with an Excel spreadsheet that includes multiple worksheets, and you want to narrow down the data source to specific worksheets, you can browse to and select specific worksheets for inclusion. To do this:

   1. Click **[...]** next to the **Worksheet filter** text box.
   2. The **Select Worksheet Filter** dialog box displays. Select the worksheets that you want included as your data source.
   3. Click **OK**.

9. *Optional:* **Key column** selection is used by tests to define which worksheet columns within a data source are used as primary key. This is helpful if your data source will undergo edits, for example when you add or remove rows within a worksheet. Even if your data source is edited, tests will still be able to identify which columns or rows should be used. Tests created from data-driven data sources use key column values in their names, rather than column numbers.

   Note: MS Excel only: If the data source includes multiple worksheets, only columns with identical names are available to be defined as key columns.

   To configure a key column:

1. Click **[...]** next to the **Key column** text box.
2. The **Select Key Column** dialog box displays. Select a column from the column list that is to act as a key column.
3. Click **OK**.

10. Click **OK** on the **New Data Source** dialog box.

Downloading Excel Files from a Data Source

Note: Files can not be downloaded from JDBC and ODBC data sources.

To download an Excel file from a data source:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Data Sources** tab. The **Data Sources** page displays, listing all of the data sources that have been created for the system.
3. Click the **Download** icon in the **Actions** column of the data source you want to download from. The **File Download** dialog box displays.
4. Click **Open** to open the file immediately, or click **Save** to specify where on your local system you want to save the file to.

Synchronizing Data Sources

You must synchronize a data source each time it is changed or updated, if you want to make SilkCentral aware of the changes. Synchronizing a data source propagates recent changes to associated tests.

To synchronize an updated data source:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Data Sources** tab. The **Data Sources** page displays, listing all of the data sources that have been created for the system.
3. Click the **Synchronize** icon in the **Actions** column that corresponds to your data source to propagate the updated file to the associated tests.
4. A confirmation dialog box displays, asking you to confirm the synchronization. Click **Yes** to synchronize all tests with the updated data source, or click **No** to abort the synchronization.

   Caution: When you synchronize a data source, all running executions depending on this data source are aborted. Results of incomplete tests within these executions are lost.
5. Click **OK** on the success message dialog box.

Uploading Updated Excel Files to a Data Source

Note: Files can not be uploaded to JDBC and ODBC data sources.

To upload an updated Excel file to a data source:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Data Sources** tab. The **Data Sources** page displays, listing all of the data sources that have been created for the system.

3. Click the **Upload** icon in the **Actions** column of the data source you want to download from.
4. Click **Browse...** on the **Upload File** dialog box.
5. Select the updated Excel file that you want to replace the currently uploaded Excel file with. Click **Open**.
6. Click **OK** on the **Upload File** dialog box.
7. A confirmation dialog box displays, asking you to confirm the overwriting of the existing file. Click **Yes** to continue.
8. After uploading the updated data source file, another dialog box displays, asking you if you want to synchronize the tests with the updated data source. Click **Yes** to synchronize immediately, or click **No** if you want to synchronize later.

   Note: After uploading an updated data source file, you must synchronize the data source so that associated tests are updated.

Deleting Data Sources

Note: Data sources that are being used by tests can not be deleted.

To delete a data source:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Data Sources** tab. The **Data Sources** page displays, listing all of the data sources that have been created for the system.
3. Click the **Delete** icon in the **Actions** column of the data source that you want to delete.
4. A confirmation dialog box displays. Click **Yes** to remove the data source, or click **No** to abort the deletion.

Data Sources Configuration Page

Project:<Project Name> > **Project Settings** > **Data Sources**

The **Data Sources** page lists all data sources that are configured for the integration of data-driven tests into Test Manager. Click **New Data Source** to create a new data source. For each data source, the page displays the following columns:

Column	Description
Name	The name of the data source as it displays in the SilkCentral GUI and in reports. Click the name to modify the data source settings.
Type	Data source type: • CSV • JDBC • MS Excel
Created On	Date when the data source was created.
Created By	User who created the data source.
Changed On	Date when the data source was last modified.
Changed By	User who last modified the data source.
Actions	You can perform the following actions on a data source: • **Delete**

Column	Description
	• **Download** • **Upload** • **Synchronize**

Requirements

This section explains how to manage requirements in Test Manager.

The **Requirements** area enables you to maintain control over system requirements during development. You can manage the creation, modification, and deletion of requirements, associate tests with requirements, track the change history, and generate test plans directly from requirement lists. As with all Test Manager functionality, the **Requirements** area is 100% Web enabled and accessible through a Web browser.

Managing Requirements

This section describes the tasks you can perform on requirements in Test Manager.

Creating Requirements

Note: Newly created Test Manager projects do not contain requirements.

To create a new requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Click **New Requirement** on the toolbar.

   Note: If the project you are working with does not yet have any requirements associated with it, click the **<Click here to add Requirements>** link in the **Requirements** tree to open the **New Requirement** dialog box.

   The **New Requirement** dialog box opens.
3. Type a meaningful **Name** and **Description** for the requirement.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for description text boxes.
4. Select the appropriate **Priority**, **Risk**, and **Reviewed** status from the list boxes.
5. If custom requirements have been defined, they are listed below the **Reviewed** list box. Type in the corresponding text box any custom property data that you want tracked with this requirement.

   Note: The **Priority**, **Risk**, **Reviewed**, and any **Custom Property** fields will be configured automatically with the corresponding properties of the parent requirement if you check the **Inherit from parent** check boxes for these properties.
6. Click **OK** to create a new top-level requirement. Alternatively, you can click **OK and New Requirement** to both save the newly created requirement and open the **New Requirement** dialog box to create an additional top-level requirement. Or, you can click **OK and New Child Requirement** to have the **New Child Requirement** dialog box open after the new top-level requirement is created.

Creating Child Requirements

The hierarchical **Requirements** tree enables you to create both top-level requirements and child requirements, which are sub-requirements of higher-level requirements. When you create a child requirement, you can specify whether the child requirement inherits the properties of the parent requirement or not.

To create a child requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement under which you want to create a child requirement.
3. Click **New Child Requirement** in the toolbar. The **New Child Requirement** dialog box opens.
4. Type a meaningful **Name** and **Description** for the requirement.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for description text boxes.
5. For each of the available requirement property fields, define whether the child requirement should inherit the values from the parent or not.

   By default, all **Inherit from parent** check boxes are checked and all parent values are inherited. To specify a property value other than that held by the parent, uncheck the corresponding **Inherit from parent** check box to unlock that property's list box or text box. Then select the specific value that the child requirement is to have. In **Details View**, asterisks (*) are placed next to requirement properties for which values have been inherited from parent requirements.

   Note: You can create child requirements at any level in the tree view hierarchy other than the top level. There is virtually no limit to the number of child requirements that can be inserted at a single hierarchy level.

Editing Requirements

To edit the properties of a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Click in the toolbar.
3. Select the requirement in the **Requirements** tree.
4. Click the **Properties** tab. The **Properties** page displays the properties of the selected tree-element.
5. Click **Edit Properties**. The **Edit Requirement** dialog box opens.

   Note: You can also access the **Edit Requirement** dialog box by clicking the **Edit** icon in the toolbar or by right-clicking the requirement in the **Requirements** tree and selecting **Edit**.
6. Edit the values displayed on the **Edit Requirements** dialog box as required. The default behavior is to inherit values from the parent requirement. Uncheck **Inherited from parent** check boxes to disable value inheritance.
7. Click **OK** to save your changes.

Note: For details regarding creating, editing, and deleting custom requirement properties, see *Custom Requirement Properties*.

Configuring Requirement Types

A requirement without a configured requirement type is not available for upload to a requirements management system (RMS). When you import a requirement from an RMS, the appropriate requirement type is automatically configured.

Note: For requirements with the type CaliberRM, Requisite Pro, and DOORS, you can only configure the requirement type for top-level tree elements, which are requirements that are a direct child of the project node. All other requirements share the requirement type of their parents.

To configure the type of a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Click in the toolbar.
3. Click the **Properties** tab. The **Properties** page displays the properties of the selected tree-element.

4. Click **Map Requirement** to select a requirement type from the list. Requirement type is a categorization used by CaliberRM, Requisite Pro, and DOORS, and is required for synchronization.

   Note: Map Requirement is only enabled when external requirements integration is enabled in the **Requirements Management** page in **Project:<Project Name>** > **Project Settings** and the requirement has not been uploaded to the external requirements management tool. Additionally, the option **Enable upload of requirements to...** must be enabled.
5. Click **OK** to save your settings and close the dialog box.

Marking Requirements as Obsolete

Rather than deleting requirements, which destroys them permanently, it is sometimes preferable to mark them as obsolete. Obsolete requirements can optionally be enabled for viewing or conversely hidden from view.

To mark a requirement as obsolete:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click ✕ in the toolbar or right-click the requirement and select **Delete**.
4. Uncheck the **Destroy permanently** check box and click **Yes**.

Note: Obsolete requirements are displayed in italics in the **Requirements** tree.

Activating an Obsolete Requirement

To convert an obsolete requirement to active status:

1. In the menu, click **Requirements** > **Details View** .
2. Right-click on the obsolete requirement in the **Requirements** tree and select **Recover**.

   Note: Obsolete requirements are displayed in italics in the **Requirements** tree.

Permanently Deleting an Obsolete Requirement

To permanently delete an obsolete requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click ✕ in the toolbar or right-click the requirement and select **Delete**.
4. Check the **Destroy permanently** check box and click **Yes**.
5. Click **Yes** on the **Delete Requirement** dialog box.

Requirements Toolbar Functions

Requirements > **Details View**

Requirements > **Document View**

The requirements toolbar provides important commands for managing your requirements.

Note: Some commands in the requirements toolbar are also available through context menus in the **Requirements** tree.

The following commands are included in the requirements toolbar:

Command	Icon	Description
Document View		Displays the **Document View**, which shows selected properties of all requirements in a single view.
Details View		Displays the **Details View**, which enables you to drill deeply into the properties of a single requirement.
New Requirement		Add a new requirement to the active project.
New Child Requirement		Add a new child requirement to the selected requirement.
Edit		Open the selected requirement for editing.
Delete		Marks the selected requirement in the tree as obsolete. The requirement will subsequently be displayed in italics if you right-click the project node and select **Show obsolete Requirements**. Check the **Destroy permanently** check box on the **Delete Requirement** dialog box to permanently delete a requirement from the system.
Cut		Cut a requirement element from the **Requirements** tree and save it to the clipboard.
Copy		Copy a requirement element from the **Requirements** tree to the clipboard.
Paste		Paste a requirement element from the clipboard to the **Requirements** tree.
Paste as Child		Paste a requirement from the clipboard as a child requirement to the currently selected requirement.
Move Up		Move a requirement up within the **Requirements** tree.
Move Down		Move a requirement down within the **Requirements** tree.
Find		Search through all requirements in the active project based on configurable parameters.
Replace		Replace instances of found values with a new value.
New Filter		Create a new custom filter for the **Requirements** tree.
Filters		Lists the available filters for the **Requirements** tree
Edit Filter		Edit the currently selected custom filter.
Delete Filter		Delete the currently selected custom filter.
Copy Filter		Copy the currently selected custom filter.
Show Changes		Show recent changes to the **Requirements** tree.
Acknowledge		Acknowledge changes to the **Requirements** tree.
Show Direct Coverage **Show Full Coverage**	 	Toggle between full and direct coverage modes. Full coverage mode offers a cumulative view of test-to-requirement coverage that considers the status of all child requirements. In direct coverage mode, the requirement status is calculated based on the tests that are assigned directly to requirements.
Download as PDF		This function is enabled only in the **Document View**. The action generates a PDF that includes all currently visible nodes of the **Requirements** tree.

Requirements Tree

Requirements are displayed, organized, and maintained through a hierarchical tree structure, the **Requirements** tree. Each node in the **Requirements** tree represents a requirement. Each requirement can have any number of child requirements associated with it. The **Requirements** tree enables you to organize requirements in any number of hierarchical levels.

Note: When the **Requirements** tree includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the tab allow you to browse through the elements included in the tree one page at a time. To display all elements as a single list, select the `[All]` link.

Collapsing or Expanding the Requirements Tree

To collapse or expand the **Requirements** tree:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement folder in the **Requirements** tree and continue with one of the following options:
   - Click ▷ to the left of the name of the requirement folder to expand the folder.
   - Click ◢ to the left of the name of the requirement folder to collapse the folder.
   - Right-click on the folder and select **Expand** or **Collapse**.

Filtering the Requirements Tree

Follow the steps below to filter the requirements tree so that only the selected node and child nodes are visible. This is useful if you want to print a sub-set of requirements to a PDF.

1. Create a custom filter that defines the requirements that you want to view.
2. Select a node in the tree view, right click and select **Filter Subtree**.

Requirements Document View

Requirements > **Document View**

The **Document View** displays the status of all tests that are assigned to the active project, including the number and percentage of `Passed`, `Failed`, `Not Executed`, and `Not Covered` tests. The **Document View** displays this coverage status information in a “heat field” chart, with the following colors:

Color	Description
Green	Passed tests.
Red	Failed tests.
Orange	Tests that are not executed yet.
Grey	Test with other status.

Requirements that are not covered by tests are listed as `Not Covered`.

The **Document View** displays all custom requirement properties as columns.

Note: Test totals accumulate to the parent level, for all currently displayed tests. If you apply a filter to the **Requirements** tree, the test totals are based on the filtered requirements only. For example, the requirement totals include tests from all child requirements that are selected with the filter, and the project totals include tests from all requirements that are selected with the filter.

The **Document View** individually saves the display settings for each user and each project.

Requirement Properties

You can add custom property fields across all requirements in the selected project in **Requirements** > **Requirement Properties**. Custom properties can subsequently be edited alongside the default properties on the **Edit Requirements** dialog box. Custom properties are displayed in **Requirements** > **Details View** > **Properties**.

For additional information on custom requirement properties, see *Custom Requirement Properties*.

Replacing Requirement Properties

In the **Requirements** area, you can locate requirements that meet certain search criteria with the **Find** command. You can also replace identified property data with alternate data that you specify with the **Replace** command. Both commands offer **Find Next** and **Find Previous** functions that allow you to step through all identified properties.

To replace a requirement property:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click on the toolbar.

   Note: This command can also be executed by right-clicking a requirement and selecting **Replace**.

   The **Replace** dialog box opens.
4. From the **Find in** list box, select a requirement property to be searched.

   This list is automatically populated with all standard property fields and any custom property fields you may have created.
5. Define your search criteria in the **Find what** portion of the dialog box.

   The **Find what** portion of the dialog box offers fields and list boxes with pre-populated values that are based on the property you select in the **Find in** list box above. The UI controls available in the **Find what** portion of the dialog box also vary based on the property type selected in the **Find in** list box. For example, selecting a custom date-type property enables one or more date fields. To specify an exact date, select **exactly** from the **Find in** list box. Then click next to the date field to specify a date using the calendar tool. Alternatively, you can select **before** or **after** from the list box. To select a date range, select **between** from the list box. Then click next to the date fields to specify start and end dates. Selecting the **Reviewed** property enables a list box from which you can select either **Yes** or **No**. Selecting the **Requirement name** or **Description** property enables a text box in which you can enter a text string.
6. In the **Replace with** text box, enter the alternate property data that you want to have replace the identified data.
7. Click **OK** to find the first instance of the property you want to replace. The first requirement that meets the search criteria will be highlighted in the tree view.
8. Click **Replace** to replace only the selected instance of the property data. Click **Replace all** to replace all instances of the property data throughout all requirements in the project.

   Note: Using the **Replace all** option will overwrite inherited requirement properties with the new value, thus removing the inheritance setting of a child requirement. Use the **Replace** option only on a parent requirement if you want the child requirements to inherit the new value.
9. Click **Find Next** on the **Find** dialog box to advance to the next requirement in the list that meets your search criteria. Or click **Find Previous** on the **Find** dialog box to return to the previous requirement in the list that meets your search criteria.

Finding Requirement Properties

In the **Requirements** area, you can locate requirements that meet certain search criteria with the **Find** command. You can also replace identified property data with alternate data that you specify with the **Replace** command. Both commands offer **Find Next** and **Find Previous** functions that allow you to step through all identified properties.

To find a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select on the toolbar to open the **Find** dialog box.

   Note: This command can also be executed by right-clicking a requirement and selecting **Find**.
3. From the **Find in** list box, select a requirement property to be searched.

   This list is automatically populated with all standard property fields and any custom property fields you may have created.
4. Define your search criteria in the **Find what** portion of the dialog box.

   The **Find what** portion of the dialog box offers fields and list boxes with pre-populated values that are based on the property you select in the **Find in** list box above. The UI controls available in the **Find what** portion of the dialog box also vary based on the property type selected in the **Find in** list box. For example, selecting a custom date-type property enables one or more date fields. To specify an exact date, select **exactly** from the **Find in** list box. Then click next to the date field to specify a date using the calendar tool. Alternatively, you can select **before** or **after** from the list box. To select a date range, select **between** from the list box. Then click next to the date fields to specify start and end dates. Selecting the **Reviewed** property enables a list box from which you can select either **Yes** or **No**. Selecting the **Requirement name** or **Description** property enables a text box in which you can enter a text string.
5. Click **OK** to begin your search. The first requirement that meets the search criteria will be highlighted in the tree view. The first requirement that meets the search criteria will be highlighted in the tree view.
6. Click **Find Next** on the **Find** dialog box to advance to the next requirement in the list that meets your search criteria. Click **Find Previous** on the **Find** dialog box to return to the previous requirement in the list that meets your search criteria.

   Note: When you reach the last or first requirement that meets the search criteria and respectively click **Find Next** or **Find Previous**, the search starts from the beginning.

Requirement Properties Page

Requirements > **Details View** > **Requirement Properties**

The **Requirement Properties** page displays high-level information about the selected requirement or project.

When requirements are synchronized with an external requirements management system, items are sometimes marked with an exclamation mark (!). This means that the marked field is not mapped to the external requirements management system. Use the property mapping feature to map property fields. If you don't use the property mapping feature, only the name and the description of the requirements are mapped. For more information, see *Editing Property Mapping*.

Item values that are marked with an asterisk (*) are values that are inherited from the parent requirement.

Click **Edit Properties** to edit the displayed properties.

The following table only applies to requirements that have a flag set.

Item	Description
Flagged By	Indicates who set the flag and when the flag was set.
Comment	Indicates the comments for the flag.
Clear Flag	Click to remove the flag.

The following table lists the properties of the selected requirement or project that are displayed on the **Properties** page.

Property	Description
Requirement Name	Name of the requirement.
Requirement ID	Identifier of the requirement.
Description	Description of the requirement.
Priority	Priority that has been configured for the requirement.
Risk	Risk that has been configured for the requirement.
Reviewed	Current review status of the requirement. `Yes` or `No`.
Custom Properties	If custom properties have been configured for the requirement, they are listed here.
Document	Source document (if any) from which this requirement was derived.
Created On	Date on which this requirement was created.
Created By	Name of the user who created this requirement.
Changed On	Date on which this requirement was last updated.
Changed By	Name of the user who last updated this requirement.

Requirement Attachments

You can upload multiple files or links as attachments to requirements or delete attachments. When you cut and paste requirements that have attachments, the attachments are automatically included with the copies.

The following attachment types are available:

- Uploaded Files (.gif, .png, .jpg, .doc, .rtf, .txt, .zip, .xls, .csv, and more)
- References to UNC paths
- References to URLs, including StarTeam URLs

Attaching a File to a Requirement

To attach a file to a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click the **Attachments** tab.

   When requirements management integration has been enabled between a Test Manager project and a CaliberRM project, the **Attachments** page includes an **Open CaliberRM** button, which enables you to manage requirement attachments directly in CaliberRM.
4. Click **Upload File** The **Upload File** dialog box opens.
5. Click **Browse** and select the file that you want to attach from your local file system.
6. Type a **Description** for the attachment.

7. Click **OK**. The attachment is uploaded to the server and associated with the selected requirement.

Note: Attaching files to a requirement may not work in Mozilla Firefox. Firefox requires usage of three slashes, for example `file:///`, for a file link, while other browsers require only two, for example `file://`. Aditionally, Firefox includes a security feature blocking links from remote files to local files and directories. For more information, see *http://kb.mozillazine.org/Firefox_:_Issues_:_Links_to_Local_Pages_Don't_Work*

Attaching a Link to a Requirement

To attach a link to a requirement

1. In the menu, click **Requirements** > **Details View** .
2. Click in the toolbar.
3. Select the requirement in the **Requirements** tree.
4. Click the **Attachments** tab.

   When requirements management integration has been enabled between a Test Manager project and a CaliberRM project, the **Attachments** page includes an **Open CaliberRM** button, which enables you to manage requirement attachments directly in CaliberRM.
5. Click **Attach Link**. The **Attach Link** dialog box opens.
6. Type the URL in the **Link** text box.
7. Type a **Description** for the attached link.
8. Click **OK**. The link is associated with the selected requirement.

Viewing a Requirement Attachment

To view a requirement attachment:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click the **Attachments** tab.

   When requirements management integration has been enabled between a Test Manager project and a CaliberRM project, the **Attachments** page includes an **Open CaliberRM** button, which enables you to manage requirement attachments directly in CaliberRM.

A list of all attachments that are associated with the requirement opens. Each attachment name serves as a link. File-attachment links open **Save As** dialog boxes, enabling you to download attachments to your local file system. Link-attachments link directly to link destinations in newly spawned browser windows.

Deleting a Requirement Attachment

To delete a requirement attachment:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click the **Attachments** tab.

   When requirements management integration has been enabled between a Test Manager project and a CaliberRM project, the **Attachments** page includes an **Open CaliberRM** button, which enables you to manage requirement attachments directly in CaliberRM.
4. Click in the **Actions** column of the attachment that you want to delete.
5. Click **Yes** on the confirmation dialog to delete the attachment from the project.

Note: Only one attachment at a time can be deleted.

Requirement Attachments Page

Requirements > **Details View** > **Attachments**

The **Attachments** page lists files and links that are attached to the selected requirement. The attachments are displayed in the order in which they are uploaded, though the list of attachments can be sorted by **Name**, **Created On**, and **Created By** properties.

Note: To display any attachments that are associated with child requirements of the selected requirement, check the **Include Child Attachments** check box.

Note: The file icons indicate whether documents are directly attached to the selected requirement, or whether they are attached to a child requirement of the selected requirement.

Single icon	The file is directly attached to the selected requirement.
Double icon	The file is attached to a child requirement of the selected requirement.

The **Attachments** page displays the following columns for each listed attachment:

Column	Description
Actions	Click ✕ to delete the attachment.
Attachment Icon	Depends on the type of the attachment.
Name	Name of the attachment.
Size	Size of the attachment.
Description	Description of the attachment.
Created On	When the attachment was created.
Created By	User who created the attachment.

Working with Tests

This section describes how to use tests in the **Requirements** area.

Assigning Tests from Grid View to Requirements

To assign one or more tests from the test area **Grid View** to one or more requirements:

1. In the menu, click **Tests** > **Details View** .
2. Click on the toolbar.
3. Select the tests that you want to assign to requirements.

   You can use your keyboard's `CTRL` and `SHIFT` keys to select multiple tests using standard browser multi-select functions.
4. Right-click the selected tests and select **Save Selection**.
5. In the menu, click **Requirements** > **Details View** .
6. Select the requirement to which you want to assign the selected tests.
7. Click the **Assigned Tests** tab.
8. Click **Assign Saved Selection**.

Note: Only tests that reside in the requirements test container are assigned. You can assign the selected tests to more than one requirement but you cannot assign them into requirements in a different project. The selection persists until you make a different selection or close Test Manager.

Assigning Tests to Requirements Manually

To manually assign tests to a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click the **Assigned Tests** tab. All tests that are available for assignment are displayed in the **Test** window.
4. If you have created a test filter, you can select it from the filter list above the **Test** window.

   To create a new test filter, navigate to the **Tests** area and click **New Filter** on the toolbar.
5. Click ⇦ left of any test you want to assign to the currently selected requirement.

   Clicking ⇦ left of a test container or test folder assigns the tests that are located in those containers or folders to the selected requirement. Tests that are located within sub-folders of those containers and folders are also assigned.

Locating Assigned Tests in the Tests Tree

To locate an assigned test in the **Tests** tree:

1. In the menu, click **Requirements** > **Details View** .
2. Click in the toolbar.
3. Select a requirement in the **Requirements** tree that has at least one test assigned to it.
4. Click the **Assigned Tests** tab.
5. In the **Actions** column of a test, click to find out in which test folder or container the test is stored in. The corresponding test folder or container is highlighted in the **Tests** window.

Removing Test Assignments

To remove a test assignment:

1. In the menu, click **Requirements** > **Details View** .
2. Click in the toolbar.
3. Select a requirement in the **Requirements** tree that has at least one test assigned to it.
4. In the **Actions** column of the assigned test you want to remove, click ✕.
5. Click **Yes** on the confirmation dialog box to confirm deletion of the assignment.

Note: To remove all test assignments from the selected requirement, click **Remove All**.

Generating Tests from Requirements Details View

You can generate tests directly out of the **Requirements** tree and assign tests to specific requirements. The **Requirements** tree serves as a template for the test-folder/test structure of the new **Tests** tree.

To generate a new test from **Details View**:

1. In the menu, click **Requirements** > **Details View** .
2. Right-click the requirement or project node that you want to convert into a test and select **Generate Test**. The **Generate Test from Requirements** dialog box displays. This dialog box enables you to specify whether the leaves, which means the lowest-level nodes, of the selected requirements sub-tree

should be converted into tests or test folders and whether the tree should be generated into a new test container or an existing container.

3. Enter a name for the new test container in the **Enter Name** text box and select a product from the **Select Product** list box to create the container within the active Test Manager project.

   The **Select Product** list box is populated with the products that are configured by a project manager. For detailed information, refer to the *SilkCentral Administration Module Help* or ask your project manager.

4. If you have defined a source control profile, select the source control profile you want to use for managing the test sources from the **Select Source Control Profile** list box.

   For detailed information on source control profiles, see *Source Control Profiles* or ask your Test Manager administrator.

5. To include all child requirements of the selected requirement in the test, check the **Include child requirements** check box.

   The check box is checked by default.

6. To have the new tests automatically assigned to the requirements from which they are created, check the **Assign newly generated Tests to Requirements** check box.

   If this option is not selected, tests must be manually associated with requirements.

   Note: This option is not available when checking **Generate Test Folders from Requirement Tree leaves**.

7. Click **OK** to create the test.

   The new test has the same structure as the **Requirements** tree.

8. A message box displays, asking if you want to switch directly to the **Tests** area. Click **Yes** to view the test in the **Tests** area, or click **No** to remain in the **Requirements** area.

Assigned Tests Page

Requirements > **Details View** > **Assigned Tests**

The **Assigned Tests** page lists all tests that are assigned to the selected requirement. If a test is assigned to more than one execution plan and the **Display in Execution Context** check box is checked, the test is displayed in a separate line of the grid for each execution plan.

All tests that are available for assignment are displayed in the right-hand **Tests** tree. To assign a test to the selected requirement, double-click on the entry in the tree or click ⇦ to the left of the entry.

If you have created a test filter, you can select it from the filter list above the **Tests** tree. To create a new test filter, navigate to the **Tests** area and click **New Filter** on the toolbar.

To automatically assign newly generated tests to the requirements from which they are generated, check the **Assign newly generated Tests to Requirements** check box on the **Generate Test from Requirements** dialog box.

The default view displays only those tests that are assigned directly to the selected requirement. Check the **Full Coverage** check box to additionally display all tests that are assigned to child requirements of the selected requirement. When **Full Coverage** is checked, tests that are not assigned to the selected requirement, but are assigned to a child of the requirement, are displayed with an overlay over the test-type icon.

If you have copied a set of tests from **Tests** > **Grid View** to the clipboard, click **Assign Saved Selection** to assign the entire set to the selected requirement. Click **Remove All** to remove all assigned tests from the requirement.

The **Assigned Tests** page displays the following columns for each listed test:

Column	Description
Actions	You can perform the following actions on an assigned test: • Click to remove the assigned test from the requirement. • Click to locate the assigned test in the right-hand **Tests** tree.
Test	Name of the assigned test. Click to view and edit the test. For configuration testing, the test is separately listed for each configuration.
Execution Plan	This column is only displayed if the **Display in Execution Context** check box is checked. Name of the execution plan that includes the assigned test. For configuration testing.
Execution Plan Parent	The configuration suite or folder in the context of which the execution plan is executed. Click to access the suite or folder in the **Execution Plans** tree. If the execution plan is not included in a configuration suite or folder, nothing is displayed.
Status	Status of the assigned test in each execution plan to which the test is assigned. To display the status of all runs of the displayed tests in each execution plan, check the **Display in Execution Context** check box. • Passed • Failed • Not Executed • N/A For tests or test packages that are included in a configuration suite, the status is an aggregation of all statuses of all configurations within the suite: • If the test is assigned only to the configuration suite, the status is `Passed` only if it is passed in all configurations. If the status is not `Passed` in all configurations, the status of the worst run is propagated, with the following top-down priority: **1.** N/A **2.** Not Executed **3.** Failed **4.** Passed The status of a requirement that is assigned to the test is `Failed` if the test fails in one or more of the configurations. • We do not recommend assigning the test to a configuration suite and one or more simple execution plans. However, in this case, the aggregate of the configuration statuses is one status, and the last statuses of the execution plans are other statuses. The status that was executed last determines the last status of the test. • If the test is assigned to a simple execution plan only, the status is updated in response to the status of the last test run.
Last Execution	Time and date of the last execution of the test.
Issues	This column is only displayed if the **Display in Execution Context** check box is not checked. Issues that are associated with this test.

Coverage

This section describes how to cover your requirements with tests.

Coverage Modes

Test Manager provides two modes for coverage. Full coverage and direct coverage. The status of a requirement in the full coverage mode considers the status of the child requirements and offers a cumulative view of the tests that cover the requirement. If one or more child requirements have the status `Not Covered`, then the full coverage status of the selected requirement is also `Not Covered`, even if the coverage status of the requirement is `Covered`. The full coverage mode enables easy evaluation of whether or not requirements are covered by tests.

The status of a requirement in the direct coverage mode is calculated based only on the status of the tests that are directly assigned to the requirement. Child requirements are not considered in calculations.

Switching Between Coverage Modes

To switch between full- and direct-coverage modes:

1. In the menu, click **Requirements** > **Details View** .
2. Click / to respectively view direct coverage or full coverage.

Coverage Page

Requirements > **Details View** > **Coverage**

The **Coverage** page, which you can only access in **Details View**, displays basic properties of the selected requirement or project. It also displays the status of all tests that are assigned to the requirement, including the number and percentage of `Passed`, `Failed`, `Not Executed`, and `Not Covered` tests. A summary of all assigned tests is listed under **Total**. Requirements that are not covered by tests are listed as `Not Covered`.

The **Details View** page displays the same coverage status information in a “heat field” chart.

To view the status of all tests that are assigned to child requirements of the selected requirement in addition to all tests that are directly assigned to the requirement, check the **Full coverage** check box.

Note: In both **Details View** and the **Coverage** page, test totals accumulate to the parent level. For example, requirement totals include tests from child requirements and project totals include tests from all requirements.

The **Coverage** page displays the following properties for each selected requirement or project:

Property	Description
Requirement Name Project Name	Name of the selected requirement or project.
Priority	Priority that has been assigned to the selected requirement.
Risk	Risk that has been assigned to the selected requirement.
# Requirements (Calc.)	Total number of all covered requirements. Not included in this number are uncovered requirements and folders that do not have a test assigned to them. For example folders that inherit the coverage of their child requirements but are not actually a requirement themselves.
# Requirements (Total)	Total number of all requirements beneath the selected entity, including folders.
Requirement Status Project Status	Status of the selected requirement or project. • `Passed` • `Failed`

Property	Description
	• `Not Executed` • `Not Covered`
# Requirements Passed	Total and percentage of requirements in the project that have tests that have passed.
# Requirements Failed	Total and percentage of requirements in the project that have tests that have failed.
# Requirements Not Executed	Total and percentage of requirements in the project that have tests that have not been executed.
# Requirements Not Covered	Total and percentage of requirements in the project that are not covered by tests.
Total	Total number of requirements in the project.

Requirement History

Test Manager provides a complete history of all changes that are made to requirements. History information is read-only, and cannot be edited or permanently deleted.

The **Recent Changes** filter, which you can access by clicking on the toolbar, enables you to efficiently view and acknowledge the latest changes and additions that have been made to requirements.

Note: When you delete a requirement with the **Destroy permanently** option, you delete a requirement that is marked as obsolete, or if you acknowledge all recent changes, a change history entry is added to the history file of the project, to which the requirements belong.

Viewing Recent Changes

To view recent changes to requirements:

1. In the menu, click **Requirements** > **Details View** .
2. Click to filter out all requirements except those that have been changed since your last change acknowledgement.

   Note: The recent changes filter is selected automatically in the **Filter** list box.
3. When you have reviewed the changes, you can accept them by clicking .

   The acknowledge function resets the recent changes filter.
4. Click again to remove filtering and see all requirements.

Tracking the History of a Requirement

To view a requirement's history:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Click the **History** tab.
4. *Optional:* When requirements management integration between a Test Manager project and a CaliberRM project is enabled, you can click **Open CaliberRM** in the **History** page to view the history of synchronized requirements directly in CaliberRM.

The properties of all revisions that have been logged by Test Manager are displayed in tabular format.

Requirement History Page

Requirements > **Details View** > **<Requirement>** > **History**

The **History** page details the revision history of the selected requirement or project. The following actions generate requirement history entries:

- Adding a new requirement
- Editing a requirement
- Marking a requirement as obsolete
- Adding an attachment to a requirement
- Deleting an attachment from a requirement
- Importing or updating a requirement through MS Word or MS Excel

For each change, the page displays the following columns:

Column	Description
Rev.	Revision number. 1-n.
Changed On	Date and time of the change.
Changed By	User that made the change.
Notes	Automatically generated description of the change. For example, “deleted” or “created”.

Note: When the page includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the page allow you to browse through the elements included on the page one page at a time. To display all elements as a single list, click the **[All]** link.

Requirement Change Notification

You can configure email notifications to alert you to changes that are made to requirement settings for specified projects. For more information on change notification, see *Change Notification*.

Requirements Reports

This section describes the requirements-related reports that ship with Test Manager. Requirements reports detail the status of functional requirements, for example compatibility requirements, GUI requirements, or feature requirements, which must be met during development. Requirements may also relate to product management objectives such as reliability, scalability, and performance. The requirement-management reports help managers to determine if adequate test coverage is established to verify that system requirements are met during development. When a report references a requirement that includes HTML-formatted content, that content is rendered in the report.

Status Reports

The following status reports are available for the **Requirements** area:

Report	Description
Requirements Status Overview	Represents a grouped summary of all requirements by current requirement coverage. Coverage is expressed by the statuses `Passed`, `Failed`, `Not Executed`, and `Not Covered`.

Report	Description
Top-Level Requirement Coverage	Represents a listing of all top-level requirements. For each requirement the number of child requirements, that are covered and not-covered by tests, is displayed.
Status of Requirements with Priority Higher than 'X'	Represents a summary of all requirements by current requirement coverage. The returned group of requirements is restricted by the `Priority` parameter, which specifies the lowest requirement priority that is considered in the data.
Requirement Impact Analysis Report	Displays the test details grouped by the execution plan that is associated to the requirement. This report allows the user to gain insight into testing assets that may be impacted by a change to the requirement. The report contains the name, status, date/time of last execution, manual testers, planned time, and issues of each test. The following statuses are used: **Status** — **Description** **Passed** — The test passed. **Failed** — The test failed. **N/A** — The test is associated to a requirement but is not submitted for execution. **Not Executed** — The test is associated to a requirement and is submitted for execution but not executed yet.
Project Requirement Traceability Report	Displays all requirements in a project along with their associated tests and issues. The following statuses are used: **Status** — **Description** **Passed** — The test passed. **Failed** — The test failed. **N/A** — The test is associated to a requirement but is not submitted for execution. **Not Executed** — The test is associated to a requirement and is submitted for execution but not executed yet.

Progress Reports

The following progress reports are available for the **Requirements** area:

Report	Description
Requirements Coverage Across Builds 'X' and 'Y'	Represents a summary of all requirements by current requirement coverage. The returned group of requirements is restricted by the `Priority` parameter, which specifies the lowest requirement priority that is considered in the data.
Requirements Coverage Over the Past 'X' Days	Represents a trend in requirements coverage by considering overall requirements coverage over 'X' days.
Specific Requirements Coverage Over the Past 'X' Days	Represents a trend in requirements coverage by considering specific requirements coverage over 'X' days.

Document Reports

The following document reports are available for requirements:

Report	Description
All Requirements	All requirements are represented with full requirement information.
Requirement with Child Requirements	The selected requirement is shown with its requirement ID. Full details regarding the child requirements of the requirement are displayed.

All Related Issues Report

The *All Related Issues* report provides a detailed list of all issues related to the assigned tests for a requirement, and explains the relationship between requirements, the assigned tests, and issues that have occurred.

Input Parameters

The input parameter for the *All Related Issues* report is the identifier of the requirement.

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Requirement Information

This section provides the following information about the requirement:

Element	Description
ID	Identifier of the requirement.
Name	Name of the requirement.
Description	Description of the requirement.
Nr. of Issues	Amount of issues related to the requirement or sub-requirements of the requirement.

Related Issues

Shows all issues related to the requirement or sub-requirements of the requirement in tabular form. The table provides the following information for each issue:

Column	Description
ID	Identifier of the issue. If an identifier is provided by the issue tracking system, this external identifier is used. The identifier is clickable if an external link is defined for the issue.
Synopsis	Meaningful short-description of the issue.
Status	Current status of the issue. If the status is provided by the issue tracking system, this external status is used.
Assigned by	Person who assigned the issue to the test.
Test ID	Identifier of the test in which the issue was discovered.
Test	Name of the test in which the issue was discovered.

Flags

Flags indicate that a requirement has changed or needs to be reviewed. Requirements with flags appear in the requirements tree with flag icons. There is also a section in the requirement properties to view flag information.

You can add and remove flags as needed and others will see the flag when they look at the requirement. You can also flag a requirement and all of its children. There can only be one flag on an item at a time and setting a flag on an item that already has a flag will overwrite the flag. Changes to the requirement flag are maintained in the requirements change history.

If you are integrating with an external requirements management tool, you will also see flags from new or updated external requirements. The following are scenarios about how flags are applied for integrated requirements:

- If you add a requirement to the requirements management tool after initial synchronization, then a subsequent synchronization is performed, when the new requirement is added to Test Manager, it will display with an *added* flag to indicate it was created during an update.
- If you edit a requirement that has been synchronized to Test Manager, the requirement in Test Manager will be flagged with a *updated* flag to indicate it was updated.

Setting a Flag

To set a flag on a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. Right click the requirement and choose **Flag** > **Set Flag** . The **Set Flag** dialog box opens.
4. Type a comment for the flag in the **Comment** text box.
5. To set the flag for all children of the currently selected requirement, click the **Set flag on child requirements** check box.
6. Click **Ok**.

Note: The flag information is available on the **Properties** page of the requirement.

Clearing a Flag

To clear a flag from a requirement:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement in the **Requirements** tree.
3. To clear a single flag: Right click the requirement and choose **Flag** > **Clear Flag** .
4. To clear the flag for all children of the selected requirement: Right click the requirement and choose **Flag** > **Clear Flag - Include Children** . The flag information is removed from the **Properties** page of the requirement.

Note: You can also clear the flag by clicking **Clear Flag** on the **Properties** page of the requirement.

Creating a Filter for Flagged Requirements

To create a filter for flagged requirements:

1. In the menu, click **Requirements** > **Details View** .
2. Click **New Filter**. The **New Filter** dialog box opens.
3. Select **Requirements Property** from the **Selection Criteria** list.

4. Select **Requirement State** from the **Property** list.
5. Select = from the **Operator** list.
6. Select one of the following values from the **Value** list:

None	No flag is selected
Flagged Manual	User manual added a flag to a requirement.
Flagged New	Requirement was created.
Flagged Updated	Requirement was updated.
Obsolete	Requirement is marked as obsolete.

7. Click **Save** and **Apply**.

Note: If child requirements are flagged, but one or more parent requirements are not flagged, the full path to the flagged child requirements will display, but with the parent requirements disabled. You cannot set or clear the flag for parent requirements that are in this state.

Requirements Import

Test Manager enables you to import requirements from Microsoft Word (Word) into Test Manager. You can import a requirement into the **Requirements** area and then maintain all included requirements in Test Manager. The import is a one-time event.

Note: Before you can import requirements, you need to create a requirements file in Word. For more information , see *Requirements File*.

Test Manager uses the information stored in the Word file and in a mapping file to determine what data to store and captures only the information that is mapped with certain styles. Test Manager uses a mapping file to map the requirements in the Word file to the Test Manager requirements. A default mapping file named `RequirementsPropertiesDefaultMapping.xml` is preselected for the import. You can choose your own customized mapping file to map your requirements. For more information about mapping files, see *Requirements Mapping File*. If you already have requirements Word documents that you used with the Office Import Tool in previous versions of Test Manager, use the mapping file `OITCompatibleRequirementsMapping.xml`.

.DOCX Functionality

The functionality for `.docx` is the same as for `.doc` with the following improvements:

- Images and any other embedded object (OLE) (for example: PDF, Excel) are detected and imported as an attachment to a requirement.
- Descriptions may contain hyperlinks.

Importing Requirements from a Word File

To import requirements from a Word file into the **Requirements** unit:

1. In the menu, click **Requirements** > **Details View** .
2. Right-click the requirement that you want to use as the parent node of the imported requirements and choose **Import from Word...**. The **Import Requirements from Word File** dialog box opens.
3. Select the mapping file from the **Choose Mapping File** list box.

   The default mapping file is `RequirementsPropertiesDefaultMapping.xml`. If you want to import requirements from Word documents that you used with the Office Import Tool in previous versions of Test Manager, use the mapping file `OITCompatibleRequirementsMapping.xml`.
4. *Optional:* Edit the requirements mapping file.
5. Either type the fully qualified path of the requirements file into the **Requirements File** text box or click **Browse** to search for the requirements file.

6. Click **OK**. A message informs you if the import was successful. Click **OK** to close the message.

Requirements Update

Test Manager enables you to update requirements that were imported from Microsoft Word (Word). You can change the requirements in the Word document and then update them in the **Requirements** area.

You can also change the mapping file to change the mapping of the requirements in the **Requirements** area.

Note: You can only update a requirement for which a unique external ID is defined in the mapping file.

Updating Requirements from a Word File

To update requirements from a Word file:

1. In the menu, click **Requirements** > **Details View** .
2. Right-click the requirement that you want to use as the parent node of the updated requirements and choose **Update from Word...**.

   Note: Update from Word... is only active if the selected requirement has been imported into Test Manager.

   The **Update Requirements from Word File** dialog box opens.
3. Select the mapping file from the **Choose Mapping File** list box.

   Note: You can only update a requirement for which a unique external ID is defined in the mapping file.

   The default mapping file is `RequirementsPropertiesDefaultMapping.xml`. If you want to import requirements from Word documents that you used with the Office Import Tool in previous versions of Test Manager, use the mapping file `OITCompatibleRequirementsMapping.xml`.
4. Either type the fully qualified path of the requirements file into the **Requirements File** text box or click **Browse** to search for the requirements file.
5. Click **OK**. A message informs you if the import was successful. Click **OK** to close the message.

Requirements File

You can create a requirements file in Word to import requirements into Test Manager. In the requirements file, you can define requirements and their hierarchy, along with their priorities, risks, review status, and custom properties.

To map the requirements in your requirements file to the requirements in Test Manager, use an XML mapping file. The mapping file maps the styles in the Word document to the requirement properties in Test Manager. You can name the Word styles as you want, as long as you map them appropriately in your mapping file.

If you want to be able to update the requirements, you must define a style for the external ID in the mapping file and assign a unique external ID to each of the requirements in the requirement file. The following example shows a sample requirement, two subrequirements, and the corresponding external IDs:

```
MyRequirement_1
  MyExternalID_1
MyRequirement_1.1
  MyExternalID_1.1
MyRequirement_1.2
  MyExternalID_1.2
```

Requirements Mapping File

The XML mapping file enables Test Manager to translate the information included in the Word requirements file into Test Manager requirements. The default mapping file is located on your front-end

server in `C:\Documents and Settings\All Users\Application Data\Silk\SCC35\OfficeImportMappingFiles\requirements\RequirementsPropertiesDefaultMapping.xml`.

The following tags map the Word styles to Test Manager requirements properties:

Title	Defines a requirement and it's level in the hierarchy. `Style name="Heading 1" level="0"` defines a top-level requirement, `Style name="Heading 2" level="1"` defines a second-level requirement, and so on.
Optional: **Description**	When a Word style is defined for this tag, all contents with the defined style are used as the description of the requirement. If no style is defined, all contents with styles that are not defined for the other tags are concatenated for the description of the requirement. The `Description` tag features basic Word text-formatting, including tables. Each image that is included in the description is attached to the requirement as an attachment.
ExternalID	Defines the external ID of the requirement, which enables the requirement to be found in the requirements tree and to be updated.
Priority	Defines the priority of a requirement. The following values are allowed: • Low • Medium • High • Critical
Risk	Defines the risk of a requirement. The following values are allowed: • Low • Medium • High • Critical
Reviewed	Defines if a requirement has been reviewed or not. Allowed values are `Yes` and `No`.
CustomProperty	Defines the custom properties that you have defined in Test Manager. You can define properties with `number`, `text`, `date`, and `list` data-types. The date format in the source document must be *dd.mm.yyyy*. For multiple-select list items, separate the items using a semi-colon without spaces. For number type items, the precision must match the precision defined in the custom property definition in **Projects:<Project Name>** > **Project Settings**. `RequirementsPropertiesDefaultMapping.xml` defines mappings for three sample custom properties: `MyInteger`, `MyString`, and `MyDate`.

The following code sample displays the contents of `RequirementsPropertiesDefaultMapping.xml`.

```
<Title>
  <Style name="Heading 1" level="0"/>
  <Style name="Heading 2" level="1"/>
  <Style name="Heading 3" level="2"/>
  <Style name="Heading 4" level="3"/>
  <Style name="Heading 5" level="4"/>
</Title>
<ExternalId>
  <Style name="ExternalID"/>
</ExternalId>
<Priority>
  <Style name="Priority" />
```

```
  </Priority>
  <Risk>
    <Style name="Risk" />
  </Risk>
  <Reviewed>
    <Style name="Reviewed" />
  </Reviewed>
  <CustomProperty propertyName="MyInteger" type="integer">
    <Style name="MyInteger"/>
  </CustomProperty>
  <CustomProperty propertyName="MyString" type="string">
    <Style name="MyString"/>
  </CustomProperty>
  <CustomProperty propertyName="MyDate" type="date">
    <Style name="MyDate"/>
  </CustomProperty>
</Mapping>
```

Editing the Requirements Mapping File

To edit the requirements mapping file:

1. In the menu, click **Requirements** > **Details View** .
2. Select a requirement from the **Requirements** tree.
3. Right-click the requirement that you want to use as the parent node of the imported requirements and choose **Import from Word...**. The **Import Requirements from Word File** dialog box opens.
4. Click **Edit Mapping**.

   The **Edit Mapping** dialog box opens.
5. Edit the variables of the requirements file in the text boxes.

   Note: For information on the variables included in the requirements file, see *Requirements File*.
6. *Optional:* Edit the name of the mapping file.
7. Click **OK** to save your changes or click **Cancel** to cancel your changes.

External Requirements Management Tools

This section explains how to work with external requirements management tools.

Synchronizing Requirements

Enabling synchronization of requirements between Test Manager and an external RMS enables Test Manager to receive changes that occur in the external RMS whenever a synchronization is executed. If a project has external RMS integration enabled, the master system for requirements is automatically the external system. This means that synchronization is always from the external RMS tool to Test Manager. Requirements can no longer be edited in Test Manager. An exception are newly created requirements that don't exist in the external tool, which are uploaded to the external (master) system only if the option **Enable upload of requirements** is enabled in **Settings** > **Requirements Management** .

Property mapping functionality allows you to map property fields between Test Manager and external requirement tools. For example, a custom field in Test Manager called `User` might be equivalent to a property field in CaliberRM called `Field_2`. The property mapping feature ensures that changes to requirement-property fields are accurately refreshed between projects. If you don't use the property mapping feature, only the name and the description of the requirements are mapped. For more information, see *Editing Property Mapping*.

Requirements can be synchronized in one of several ways:

Manual synchronization	Click **Synchronize Changes** at the root folder level on the **Properties** page to synchronize only requirements that have changed since the last synchronization. This option is not available for IBM Rational RequisitePro.
Forced manual synchronization	Click **Synchronize All** at the root folder level on the **Properties** page to force synchronization of all requirements.
Automatic scheduled synchronization	Based on globally defined Test Manager schedules.
Automatic online synchronization	Changes to requirements are automatically propagated between tools. This option is available for CaliberRM only. It requires CaliberRM client installation on the application server and MPX enabled. To enable automatic synchronization between CaliberRM and Test Manager, add the file `ss.jar` to the application server at `Program Files\Silk\SC Test Manager <version>\lib` and restart the application server. Contact customer care to obtain the file. Requirement data is automatically updated in Test Manager when changes are made in CaliberRM and traces in CaliberRM are updated when test assignment changes are performed in Test Manager. This type of online synchronization is only available when projects are configured with the current baseline.

Automatic synchronization of requirements between Test Manager and external requirements management tools can be configured to occur based on global schedules. For details on configuring global schedules, refer to the *SilkCentral Administration Module Help*.

Note: The **Open CaliberRM** buttons open whatever program is registered as the default program for opening files of extension `.crm`. On some machines, this may be the requirement viewer, rather than CaliberRM. This behavior can be changed by your administrator. The client program is called `caliberrm.exe`. When properly configured, the program opens to the requirement that is selected in Test Manager.

The binder icon on the project node of the **Requirements** tree indicates the status of RM integration for the project:

No configuration	RM integration is not available.
Manual configuration	Requirement import, upload, and synchronization can be done only by clicking the corresponding buttons on the project node in **Requirements** > > **Properties**.

At the project level, the **Properties** page includes the following properties:

Property	Description
System	The external tool with which integration has been enabled.
Status	Whether or not integration has been enabled.
Project Name	The name of the external project that the Test Manager project is associated with.
Requirement Types	The requirement types that are shared between projects.
Last Synchronization	Date and time of the last synchronization.
Last Synchronization Status	The status of the last synchronization including the number of created, updated, and deleted items.

Note: When integration between CaliberRM and Test Manager with automatic online synchronization has been enabled, the project node displays the current status of the online requirements change listener. The three possible statuses for such projects are: `Connected` (synchronized), `Reconnected` (synchronization recommended), and `Disconnected`.

Synchronizing Requirements Across Tools

Note: Use the property mapping feature to map property fields. If you don't use the property mapping feature, only the name and the description of the requirements are mapped. For more information, see *Editing Property Mapping*.

To synchronize requirements between Test Manager and an externally configured requirements management tool

1. In the menu, click **Requirements** > **Details View** .
2. Select the Project node in the **Requirements** tree.
3. Click the **Properties** tab. The **Properties** page displays the properties of the selected tree-element.
4. Click **Synchronize Changes**.
5. Click **Yes** on the **Synchronize Requirements** confirmation dialog box to begin synchronization. A dialog box opens when synchronization is complete, displaying synchronization statistics, including the number of requirements that have been created, updated, and deleted.
6. Click **OK** to complete the synchronization. Any updates that were made to mapped requirements in your externally configured requirements management tool are now reflected in the **Requirements** tree.

Synchronizing Requirements based on Schedules

You can configure global schedules to automatically synchronize requirements between Test Manager and an external requirements management tool. To enable automatic synchronization between CaliberRM and Test Manager, add the file `ss.jar` to the application server at `Program Files\Silk\SC Test Manager <version>\lib` and restart the application server. To obtain the file, contact customer care.

To synchronize requirements based on globally defined schedules:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Requirements Management** tab.
3. Click **Edit Schedule**. The **Edit Schedule** dialog box opens.
4. Click the **Global** option button.
5. Select a pre-defined global schedule from the selection list.

   Note: For details about configuring a global schedule, refer to the *SilkCentral Administration Module Help*.
6. Click **OK**.

Defining Email Notification for Automatic Synchronization Events

You can define an email notification to alert users when errors occur during automated synchronization of requirements between Test Manager and external requirements management tools. All notification recipients receive copies of the synchronization log files.

To define an email notification for automatic synchronization events:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Requirements Management** tab.
3. Click **Edit Notification**. The **Edit Notification** dialog box displays.
4. Check the **Enable notification** check box.
5. Select a user name from the **Username** list box.

Note: For details about defining users, refer to the *SilkCentral Administration Module Help*.

6. If required, add additional email addresses for other recipients in the **Other email addresses** text box. Use semicolons to separate multiple email addresses.
7. Click **OK**.

CaliberRM Integration

This section describes how to integrate CaliberRM with Test Manager.

Note: CaliberRM user defined attribute **Multiple selection user list** and **Multiple selection group list** must be mapped to the Test Manager text attribute type and not list type.

Enabling Integration with CaliberRM

To enable integration with CaliberRM:

1. In the menu, click **Projects** > **Project List**.
2. Select the project to which you want to establish integration.
3. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
4. Click the **Requirements Management** tab.
5. In the **Borland® CaliberRM Integration** section, click **Configure**. The **Edit Configuration** dialog box opens.
6. Type the **Hostname** of the machine where the external server is installed.
7. Type valid **Username** and **Password** credentials for the requirements management server.
8. Click **Test Connection** to confirm that the host and user credentials you have entered are correct.

   Click **Browse** to advance to the **Browse Projects** dialog box.

   If the settings are correct, a **Test connection was successful** dialog box opens.

   Note: If you are not able to establish a connection, consult your system administrator.
9. Click **OK**.
10. From the **Project** list box, select the external project with which the Test Manager project is to be integrated. The requirement types that are available with the selected project are automatically populated into the **Requirement Types** field. The baselines that are available with the selected project are automatically populated into the **Baseline** field.
11. Select a **Baseline** from the external project that should be integrated with the Test Manager project.

    Your selections are displayed on the **Edit Configuration** dialog box.
12. Click **OK**.
13. Select one or more requirement types from the external project that should be integrated with the Test Manager project.

    Use `CTRL + Click` to select multiple requirement types.
14. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with CaliberRM.
15. *Optional:* Check the **Enable upload of requirements to CaliberRM** check box to enable the upload of unmapped or unassigned requirements from Test Manager to CaliberRM. This allows you to upload additional previously unmapped requirement trees to CaliberRM and then have those requirements mapped within Test Manager. The **Map Requirement** button in **Test Manager** > **Requirements** > **Properties** becomes enabled, allowing configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.

16. Click **OK** to save your settings.

Baseline Support for CaliberRM Integration

You can select the current baseline or existing user-defined baselines for CaliberRM integration. When you select a user-defined baseline, the **Map Requirement** button in **Requirements** > **Details View** > **<Requirement>** > **Properties** is disabled and requirements are not updated in the integrated CaliberRM project.

You cannot import modified baselined requirements into Test Manager. Requirements that are not of the current baseline can only be changed in CaliberRM if the version of the requirement that is used for the baseline is changed. Such changes are only updated within Test Manager requirements when a manual or scheduled synchronization is performed.

You can change a baseline after you import it into Test Manager. You can change the configured baseline to a different user-defined baseline or the current baseline. After such a change, the next synchronization of the baseline, either manual or scheduled, will update the Test Manager project and update, create, or delete requirements as required. When a baseline is changed, a message displays that states that the changes will take effect after the next synchronization. When a baseline is changed from the current baseline to a user-defined baseline, a message displays informing you that, for user defined baselines, upload of requirements is disabled.

Note: With user-defined baselines, you can use the same requirement type of a CaliberRM project more then once. With current baselines, a requirement type of a CaliberRM project is allowed only once.

Handling Test Assignments in CaliberRM

Assigned tests are displayed, managed, and created as traces (“Trace to”) of synchronized requirements in CaliberRM.

When synchronized requirements are assigned to tests in Test Manager, the test assignments are pushed out to and displayed within CaliberRM. Conversely, when tests are assigned to synchronized requirements from CaliberRM, the assignments are pushed out to and displayed within Test Manager.

Copying CaliberRM-Integrated Projects

To manage CaliberRM baselines when copying Test Manager projects:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Requirements Management** tab.
3. Verify that the baseline you want to save is selected.

   Note: If a baseline is changed, you must perform a synchronization to update the project requirements with the baseline changes, before you can copy an associated Test Manager project. The integration configuration is only copied if you select a baseline other than the current baseline. If you select the current baseline, you need to specify if you want to keep the integration configuration in the original project or move it to the copied project.
4. If the baseline that you want to save is not selected, click **Edit Configuration**. The **Edit Configuration** dialog box displays.
5. Click **Browse** next to the **Project name** text box. The **Browse Projects** dialog box opens.
6. Select the baseline you want to save, then confirm your selection.
7. In the menu, click **Projects** > **Project List** . The **Projects** page displays, listing all existing projects and project baselines.
8. Click in the **Actions** column of the project you want to copy.

Note: For full details on copying a project, refer to the *SilkCentral Administration Module Help*.

The **Copy Project** dialog box displays.

9. Select the items you want to copy into the new project, then confirm your selection.
10. Apply the baseline that you want to continue working with to the Test Manager project.

Note: After copying a project, the original project and the copy are identical. Define on which you will continue working on by applying the corresponding baseline.

Creating a Baseline for a CaliberRM-Integrated Project

To create a baseline of a CaliberRM-integrated Test Manager project:

1. Create a baseline of the Test Manager project.

   For more information on creating a baseline, refer to the *SilkCentral Administration Module Help*.

   A SilkCentral message box displays during the process, asking if you want to enable the CaliberRM integration for the new baseline.
2. Click **Yes**. When the new baseline is created, the **Baseline Project - Adapt Project Settings** dialog box opens.
3. Click **Edit** to change the requirements integration settings.

   The **Edit Configuration** dialog box opens.
4. Click **Browse** next to the **Project name** text box. The **Browse Projects** dialog box opens.
5. Select a CaliberRM project, a baseline, and the requirement type for synchronization.
6. Click **Ok** to close the **Browse Projects** dialog box.
7. Click **Ok** to close the **Edit Configuration** dialog box.
8. Click **Finish** to close the **Baseline Project - Adapt Project Settings** dialog box.

Enabling Integration with IBM Rational RequisitePro

To enable integration with IBM Rational RequisitePro:

1. In the menu, click **Projects** > **Project List**.
2. Select the project to which you want to establish integration.
3. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
4. Click the **Requirements Management** tab.
5. In the **IBM® Rational® RequisitePro® Integration** section, click **Configure**. The **Edit Configuration** dialog box opens.
6. Type the **UNC project path** to the machine where the external server is installed. Or click **Browse** and select the path.
7. Type valid **UNC Username** and **UNC Password** credentials for the machine where the external server is installed.
8. Type valid **Username** and **Password** credentials for the requirements management server.
9. Click**Test Connection** to confirm that the host and user credentials are correct. If the settings are correct, a **Test connection was successful** message displays.
10. Click **OK**.

    Note: Consult your system administrator if you are not able to establish a connection.
11. Click **Edit Packages and Requirement Types**.

    Your selections are then displayed on the **Edit Configuration** dialog.

Note: Only requirements of explicitly selected packages will be synchronized. Selecting a parent package does not select the child packages of the parent.

The **Browse Packages & Requirement Types** dialog box opens. The packages and requirement types that are available with the selected project are automatically populated into the **Packages** and **Requirement Types** list boxes.

12. From the **Packages** list box, select one or more packages from the external project that should be integrated with the Test Manager project.

    Use `CTRL + CLICK` to select multiple packages.

13. From the **Requirement types** list box, select one or more requirement types from the external project that should be integrated with the Test Manager project.

    Use `CTRL + CLICK` to select multiple requirement types.

14. Click **OK**.The **Browse Packages & Requirement Types** dialog box closes.
15. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with RequisitePro.
16. *Optional:* Check the **Enable upload of requirements to RequisitePro** check box to enable the upload of unmapped or unassigned requirements from Test Manager to RequisitePro.This allows you to upload additional previously unmapped requirement trees to RequisitePro and then have those requirements mapped within Test Manager. The **Map Requirement** button in **Test Manager** > **Requirements** > **Properties** becomes enabled, allowing configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
17. Click **OK** to save your settings.

Telelogic DOORS Integration

This section describes how to integrate Test Manager and Telelogic DOORS® (DOORS).

Installing Telelogic DOORS on the Front-End Server

To integrate Test Manager and Telelogic DOORS, install the DOORS client on the Test Manager front-end server machine. If you use more than one front-end server machine, you must install the DOORS client to the same directory on each of the machines.

To install the DOORS client on the Test Manager front-end server machine:

1. In the menu, click **Help** > **Tools** .
2. Click **Test Manager Add-In for Telelogic DOORS** to download the DOORS plug-in package.

   The package contains two ZIP-Archives:

   - `DoorsRM.zip`
   - `DoorsClientLibs.zip`

3. Create a new folder with the name `testmanager` in the `…\lib\dxl` folder of your Telelogic DOORS client installation.

   The default path for this folder is `C:\Program Files\Telelogic\DOORS_8.20\lib\dxl\testmanager`.

4. Extract all DOORS script files from `DoorsClientLibs.zip` to this folder.

The plug-in package `DoorsRMPlugin.zip` is automatically installed to the `Plugins` folder of your Test Manager application server installation during the setup process. During startup of the application server, this plug-in will be published to all front-end servers.

Configure a Project for Integration with DOORS

To configure a project for requirements integration with DOORS:

1. From the Test Manager project to which you want to establish integration, click **Project:<Project Name>** > **Project Settings** > **Requirements Management** .
2. In the **Telelogic DOORS® Integration** section, click **Configure**.

   The **Edit Configuration** dialog box opens.
3. In the **RM service URL** text box, type the URL of the Test Manager DOORS requirement Web Service.

   The default value should point to the correct location already. For example `http://MySCTMHost:19120/services/doorsrequirementsmanagement`.
4. Type valid **Username** and **Password** credentials for the requirements management server.
5. The default DOORS client installation path is displayed in the **DOORS Installation Path** text box. If this path is not correct, click **Browse** to browse to and select the correct destination in the front-end server directory structure.
6. Click **Test Connection** to confirm that the host and user credentials you have entered are correct.

   A **Connection successful** message box displays if the settings are correct. Click **OK** to continue.

   Note: Consult your system administrator if you are not able to establish a connection.
7. Click **Browse** next to the **Project name** text box to open the **Browse Requirement Types** dialog box. From the **Project** text box, select the external project with which the Test Manager project is to be synchronized.

   The requirement types that are available with the selected project are automatically populated into the **Requirement types** text box. Select the requirement types that are to be synchronized and click **OK**. Use `CTRL + CLICK` to select multiple requirement types.
8. Your selections are now displayed on the **Edit Configuration** dialog box. Click **OK**.
9. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with Telelogic DOORS.
10. *Optional:* Check the **Enable upload of requirements to Telelogic DOORS®** check box to enable the upload of unmapped or unassigned requirements from Test Manager to DOORS.This allows you to upload additional previously unmapped requirement trees to DOORS and then have those requirements mapped within Test Manager. The **Map Requirement** button in **Test Manager** > **Requirements** > **Properties** becomes enabled, allowing configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
11. Click **OK** to save your settings.

    Caution: As the DOORS application object is used for communication, and this object does not support login data, but rather requires a running DOORS client, Test Manager starts each DOORS client process with the provided login data and then uses that same data for all subsequent application objects. Therefore only one set of DOORS login credentials is supported for communication at one time. It is recommended that you use the same DOORS credentials for all configurations so that integration tasks can be performed on the front-end server for all projects at the same time. When a second set of credentials is used, the second set only works after all sessions using of the first set of credentials have timed out.

Enabling Integration with CaliberRDM

To enable integration with CaliberRDM:

1. From the Test Manager project to which you want to establish integration, click **Project:<Project Name>** > **Project Settings** > **Requirements Management** .
2. In the **CaliberRDM Integration** section, click **Configure**.

   The **Edit Configuration** dialog box opens.
3. Enter the **Hostname** of the machine where the external server is installed.
4. Enter the **Port** number on which the external server is listening.

5. Enter valid **Username** and **Password** credentials for the requirements management server.
6. Click **Test Connection** to confirm that the host and user credentials you have entered are correct.

   A **Connection successful** message box displays if the settings are correct. Click **OK** to continue.

   Note: Consult your system administrator if you are not able to establish a connection.
7. From the **Project name** text box, select the external project with which the Test Manager project is to be integrated.
8. Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with CaliberRDM.
9. Click **OK** to save your settings.

Rally Integration

Integrating the project management tool Rally enables you to define user stories which you can then use as requirements in Test Manager. You can then create tests in Test Manager to cover these requirements. To update the tests you have created with any changes to the corresponding user stories in Rally, synchronize the requirements.

The following user story properties are synchronized from the Rally REST Service. You can use these properties to help filter and categorize the user stories.:

Property	Description
Name	The name of the user story in Rally.
Rally ID	The identifier of the user story in Rally. This property is the External ID of the requirement in Test Manager.
Description	The description of the user story in Rally.
Iteration	The iteration in Rally in which the user story is included.
Project	The project in Rally in which the user story is included.
Release	The project release in which the user story is included.
State	The state of progress of the user story in Rally. The following states are available: • `Defined` • `In-Progress` • `Completed` • `Accepted`
Plan Estimate (Story Points)	The estimated time in Rally for the user story to be completed. The time is estimated in story points.

Enabling Integration with Rally

To enable integration with Rally:

1. In the menu, click **Projects** > **Project List**.
2. Select the project to which you want to establish integration.
3. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
4. Click the **Requirements Management** tab.
5. In the **Rally Integration** section, click **Configure**. The **Edit Configuration** dialog box opens.
6. Type the **URL** of the machine where the Rally REST Service is installed.

By default, the URL is set to `https://rally1.rallydev.com/slm/webservice/1.22/`.

7. Type valid **Username** and **Password** credentials for the Rally REST Service.
8. *Optional:* If direct access to the Internet is restricted, and the Rally REST service is located in a machine outside the scope of the internal network, use the **Proxy Host** and **Proxy Port** text boxes to specify a proxy through which Test Manager can connect to Rally .

   Note: To connect to Rally by using a proxy, you need to fill out both text boxes, **Proxy Host** and **Proxy Port**.
9. Click **Test Connection** to confirm that the host and user credentials you have entered are correct. If the settings are correct, a **Test connection was successful** dialog box opens.

   Note: If you are not able to establish a connection, consult your system administrator.
10. Click **OK**.
11. Click **Load**. The **Project** list box is populated with all the projects from all workspaces in Rally that you have permissions for.
12. From the **Project** list box, select the external project with which the Test Manager project is to be integrated.
13. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with Rally.
14. Click **OK** to save your settings.

Generating Tests from External Requirements

From the RMSs currently shipped with Test Manager, only CaliberRDM supports generating tests. If you want to use another RMS to generate tests, the RMS must be able to generate tests. The Test Manager plug-in API includes a method to verify that the RMS is able to create tests. For more information, see *Requirements Plug-In API Interfaces*.

To generate tests from external requirements:

1. Establish integration with the external requirement system.

   For additional information, see *Enabling Integration with CaliberRDM*.
2. Select the requirement in the **Requirements** tree from which you wish to generate tests.

   Note: To be able to generate tests, the RMS plug-in needs to implement the `RMTestProvider` interface, and the interface method `isTestGenerationSupported` needs to return `true` for the type of the selected requirement.
3. Click **Generate Tests**. The **Generate Tests** dialog box displays.
4. Use **SHIFT + CLICK** or **CTRL + CLICK** to select the tests that you want to generate.
5. Click **Generate All** to generate all tests, or click **Generate Selected** to generate the selected tests. The **Tests** tree displays.
6. Select the folder to which you want to add the generated tests.

   Note: Tests that already exist in the selected destination folder are not created, but updated. No tests in the folder are deleted.
7. Click **OK**.

Working with External Properties

This section explains how to work with external properties in Test Manager.

Editing External Properties

To edit external properties:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement for which you intend to edit external properties.
3. Click the **Properties** tab. The **Properties** page displays the properties of the selected tree-element.
4. Click **Edit External Properties**.

   The **Edit External Properties** dialog box displays. All properties of the external requirement are displayed here. Edit all properties as required.

   Note: Editable properties on this dialog box offer input fields and controls with which you can edit the properties. If a mapping rule exists for an attribute, the attribute will be tagged with a trailing asterisk (*).
5. Click **OK** to save your changes and close the dialog box.

Viewing External Properties

To view external properties:

1. In the menu, click **Requirements** > **Details View** .
2. Select the requirement.
3. Click the **Properties** tab. The **Properties** page displays the properties of the selected tree-element.
4. Click **View External Properties**. The **View External Properties** dialog box displays. All properties of the external requirement are displayed here.
5. Close the dialog box.

Editing Property Mapping

The property-mapping functionality allows you to map property fields between Test Manager and external requirements-management tools. For example, a custom requirement property in Test Manager called `User` might be equivalent to a custom property in CaliberRM called `User_ID`. The property-mapping feature ensures that requirement-property fields are accurately populated between projects during requirement uploading and importing. If there are multiple requirement types, you must map each type separately.

Note: If you don't use the property mapping feature, only the name and the description of the requirements are mapped.

To edit property mapping:

Note: The following applies for synchronizing properties with boolean values: since Test Manager does not support the boolean data type, you need to create a single select list with the following two list items:

- Name = `Yes`, Numeric Weight = `1`
- Name = `No`, Numeric Weight = `0`

Note: When you map lists, requirement properties from both products need to have the same named options for each list item, and they are case sensitive.

1. Select the project.
2. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
3. Click the **Requirements Management** tab.
4. Click **Edit Property Mapping** for the configured external tool. The **Edit Property Mapping** dialog box opens.
5. Select an external requirement type from the **Requirement types** list. All custom requirements of that type are then displayed below in the selection box.

6. Select the custom requirement property for which you are establishing mapping.
7. From the list box on the right, select the Test Manager custom property to establish mapping to the external custom property you have selected.
8. Click **Add Mapping** to map the requirements. The results are displayed in the **Custom property mapping** box.
9. The **System property mapping** box displays the two pre-configured mappings for requirement name and description, which cannot be removed.
10. Click **OK** to save your changes.

Deleting Property-Mapping Value Pairs

To delete a property-mapping value pair:

1. Select the project.
2. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
3. Click the **Requirements Management** tab.
4. Click **Edit Property Mapping** for the configured external tool. The **Edit Property Mapping** dialog box opens.
5. Select the property-mapping value pair in the **Custom property mapping** select box.
6. Click **Remove Mapping**.
7. Click **OK** to save your changes.

Disabling Requirements-Management Integration

To disable requirements-management integration configuration:

1. Select the project.
2. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
3. Click the **Requirements Management** tab.
4. Click the **Disable Configuration** button of the requirements-management tool for which you want to disable integration.

All integration data and functionality is disabled, but not deleted from the project.

Removing Requirements-Management Integration

To remove requirements-management integration:

1. Select the project.
2. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
3. Click the **Requirements Management** tab.
4. Click **Remove Configuration** of the requirements-management tool for which you want to remove integration.

   This button is only enabled if the configuration is disabled.

   The **Remove External Association** dialog box displays.
5. Click **Yes**.

All related data is removed from the database.

Tests

This section explains how to manage tests in Test Manager.

The **Tests** area enables you to maintain control over tests across the system development life-cycle. You can create, schedule, and manage both automated tests and manual tests. You can upload and associate files and links as attachments to test containers and tests. You can easily associate found issues with the tests that led to their discovery. You can also track the full change history of the test in the area.

Tests Document View

Tests > **Document View**

The **Document View** provides a high-level view of the important information, the status, and the last executions of all tests in the selected project.

For each test container, test folder, and test, the **Document View** displays the following columns:

Item	Description
Status	The status of the last test execution. • Passed • Failed • Not Executed • N/A For test containers and folders, a status bar displays the number of tests with the corresponding status.
Tests	Number of tests .
Planned Time	Roll up of **Planned Time** of the manual tests.
Assigned Issues	Number of Issues assigned to the test.
Last Execution	Time of the last execution of the selected test or project.
Last Build	Build on which the last execution was performed.
Changed On	Last time the selected test element was changed.
Changed By	User that changed the selected test element for the last time.

Working with the Tests Tree

Tests > **Details View**

As with requirements, tests are displayed, organized, and maintained through a hierarchical tree structure, the **Tests** tree. The **Tests** tree enables you to organize tests in any number of hierarchy levels. Each node in the tree represents either a test, a test folder, or a test container.

In the **Contents** page you can view, cut, copy, and paste the child elements of any selected test element. The page supports standard Windows Explorer style multi-select functionality.

Note: When the **Tests** tree includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the tab allow you to browse through the elements included on the tab one page at a time. To display all elements as a single list, click the **[All]** link.

Expanding or Collapsing the Tests Tree

You can consolidate levels of the **Tests** tree or display all levels of the tree based on your viewing needs.

To collapse or expand levels of the **Tests** tree:

1. In the menu, click **Tests** > **Details View** .
2. Select the test container or test folder in the **Tests** tree and continue with one of the following options:
   - Click ▷ to the left of the name of the test element to expand the element.
   - Click ◢ to the left of the name of the test element to collapse the element.
   - Right-click on the element and select **Expand** or **Collapse**.

Cutting, Copying, Pasting, and Deleting Test Elements

The toolbar in the **Tests** area enables you to easily cut, copy, paste, and delete test elements within the **Tests** tree, both within the current project and between projects. These editing functions simplify the process of building and managing the tests of your project.

The data types that are copied along with tests and test folders are properties, attributes, parameters, and attachments. Assignments, issues, runs, and history are not copied.

Tip: In the **Contents** page, you can view, cut, copy, and paste the child elements of any selected test element. The page supports standard Windows Explorer style multi-select functionality. Before you can paste a test element into the **Contents** page you must explicitly select an element within the page to gain the application's focus.

Note: Containers cannot be copied or pasted.

To cut, copy, paste, or delete a test element:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the container, folder, or test that you wish to edit.
3. Click the appropriate toolbar button:

Button	Description
✕	Deletes the selected element from the tree.
✂	Cuts the selected element from the tree and moves it to the clipboard.
⎘	Copies the selected element to the clipboard.
📋	Pastes a copy of the element held on the clipboard to the same level of the currently selected element.
📋	Pastes a copy of the element held on the clipboard as a sub-node of the currently selected element.

Note: The commands are also available through the context menu in the **Tests** tree.

Copying and Pasting a Test Element from One Project to Another

To copy and paste a test folder or test between projects:

1. In the menu, click **Projects** > **Project List**.
2. Select the source project.
3. In the menu, click **Tests** > **Details View** .
4. Click ✂ to cut or ⎘ to copy the element to the clipboard.

5. In the menu, click **Projects** > **Project List**.
6. Select the destination project.
7. Select the destination container and folder.
8. Click .

Note: The commands are also available through the context menu in the **Tests** tree.

Reordering Test Elements

To reorder a test, test folder, or test container:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test element that you wish to reorder.
3. Click to move the test element up one step or click to move the test element down one step.

Setting a Test Node as Integration Default for External Agile Planning Tools

To use the Web service calls to create tests in Test Manager through an external agile planning tool, set a folder or container in the **Tests** tree as the integration default node, where the Web service will create the test. If you do not specify the integration default node, an error message box displays.

To specify the integration default node in the **Tests** tree:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the folder or container which you want to set as the integration default node.
3. Right-click the folder or container and select **Set as Integration Default**.

   Note: If an integration default node already exists, the default node is changed to the new node.
4. The integration default node is set to the selected node, enabling the agile planning tool to create tests at this location.

   Note: The integration default node is shown in the **Properties** page of the project, in which the node is located.

The integration default node is set to the selected node, enabling the agile planning tool to create tests at this location.

Note: The integration default node is shown in the **Properties** page of the project, in which the node is located.

Filtering on a Folder or Container

To filter on a folder or container:

1. In the menu, click **Tests** > **Details View** .
2. Right-click the folder or container in the tree that you want to filter and select **Filter Subtree**. The tree displays only the contents of the selected folder or container and the hierarchy up to the root node of the tree.

Note: To remove filtering and display all elements, select **<No Filter>** from the **Filter** list box on the toolbar. Empty folders are not shown in the filtered sub-tree.

Test Properties Page

Tests > **Details View** > **Properties**

The **Properties** page offers detail on all properties and relevant information for the selected test, test folder, test package, or test container. For test nodes, the following properties are configured when tests are created:

Property	Description
Test Name	Name of the test.
Test ID	Database identifier of the test.
Description	Description of the test. **Note:** Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
Status	For tests or test packages that are included in a configuration suite, the status is an aggregation of all statuses of all configurations within the suite: • If the test is assigned only to the configuration suite, the status is `Passed` only if it is passed in all configurations. If the status is not `Passed` in all configurations, the status of the worst run is propagated, with the following top-down priority: **1.** N/A **2.** Not Executed **3.** Failed **4.** Passed The status of a requirement that is assigned to the test is `Failed` if the test fails in one or more of the configurations. • We do not recommend assigning the test to a configuration suite and one or more simple execution plans. However, in this case, the aggregate of the configuration statuses is one status, and the last statuses of the execution plans are other statuses. The status that was executed last determines the last status of the test. • If the test is assigned to a simple execution plan only, the status is updated in response to the status of the last test run.
Last Execution	Last time this test was executed. For tests that are part of a running execution plan, the last execution is updated based on the current test run.
Assigned Executions	Lists all execution plans that the selected test is assigned to. For each execution plan in the list, the grid displays the following columns: **Column** — **Description** **Execution** — Name of the execution plan. Click to view or edit the execution plan. **Status** — The status of the last run of the test in the execution plan. The status is `Not`

Property	Description
	Column / **Description** `Executed`, if the test was not executed in the last run of the execution plan. For data-driven tests or test packages, the status of all last runs is displayed in a bar graph. **Last Execution**: Last time the test was executed as part of the execution plan. **Execution Plan Parent**: The configuration suite or folder in the context of which the execution plan is executed. Click to access the suite or folder in the **Execution Plans** tree. If the execution plan is not included in a configuration suite or folder, nothing is displayed.
Created On	Date and time the test was created.
Created By	Name of the user who created the test.
Changed On	Date and time the test was last changed.
Changed By	Name of the user who last changed the test.
Planned Time [hh:mm]	Planned execution time of the test. Only for manual tests.
Test Properties	Test properties that are specific to the test type.
Success Conditions	All success conditions that have been configured for the test. For test package nodes, all success conditions except the execution time-out are disabled and hidden.
Integration Default Folder	Shows the name of the default container or folder, where tests from external RMSs are created.

Test Contents Page

Tests > **Details View** > **Contents**

The **Contents** page displays the child elements of the selected project, test container, or folder in the **Tests** tree. The page supports Standard Windows Explorer style multi-select functionality.

Tip: To drill down into the selected folder or container, press **Enter** or double-click the selected item. Press **Backspace** or click on the toolbar to navigate one level up.

Note: You cannot copy or paste a test container.

For each child element of the selected project, test container, or folder, the **Contents** page displays the following columns:

Column	Description
Name	Name of the test element.
Changed On	Date the test element was last edited.
Changed By	User who last edited the test element.

Tip: As with test elements listed in the **Tests** tree, you can right-click the elements listed on the **Contents** page to access context-relevant commands through a context menu. Commands that are not available are grayed out. Before you can paste a test element into the **Contents** page you must explicitly select an element within the page to gain the application's focus.

Note: When the **Contents** page includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the page allow you to browse through the elements included on the page one page at a time. To display all elements as a single list, click the **[All]** link.

The **Contents** page supports the following keyboard functions (shortcuts) for test elements:

Key	Normal	SHIFT	CTRL
Up	Move selection up	Extend selection up	Move up
Down	Move selection down	Extend selection down.	Move down
Left	Deselect		
Right	Deselect		
A			Select All
X			Cut
C			Copy
V			Paste
Pos1	Select first item	Select up to first item	
End	Select last item	Select down to last item	
Ins	Insert		
Del	Delete		
F2	Edit		

Keyboard Functions for Test Elements

The following mouse and keyboard combination functions are available. Following these functions, actions like cut, copy, or paste can be performed on the selected nodes:

Keyboard Function	Description
Click	Select a row and remember it as the current row.
Ctrl+Click	Toggle the selection status of the clicked row and remember it as the current row.
Shift+Click	Select the span from the currently-selected row to a newly selected row.
Ctrl+Shift+Click	When a row is already selected, this function adds the span from the current row to the clicked row to the selection. If no row is currently selected, this function removes the span from the current row to the clicked row from the selection and selects the clicked row.

Note: Containers cannot be copied or pasted.

Test Toolbar Functions

Tests > **Details View**

The test toolbar provides important commands for managing your tests.

Note: The commands in the test toolbar are also available through context menus in the **Tests** tree.

The following commands are included in the test toolbar:

Command	Icon	Description
Document View		Displays the **Document View**, which shows selected properties of all test elements in a single view.
Details View		Displays the **Details View**, which enables you to drill deeply into the properties of a single test.
Grid View		Displays the **Grid View**, which shows all tests in a grid.
Up		Navigates one level up in the hierarchy of the navigation tree, regardless of the current cursor focus.
New Test Container		Enables creation of new test containers.
New Child Test Folder		Enables creation of new test folders as sub-nodes of the selected folder or container.
New Child Test		Enables creation of new tests as sub-nodes of the selected folder or container.
Edit		Edit the selected test element.
Create Version		Create a new version of the selected manual test.
Delete		Delete the selected test element.
Cut		Cut a test element from the **Tests** tree and save it to the clipboard.
Copy		Copy a test element from the **Tests** tree to the clipboard.
Paste		Paste a test element from the clipboard to the **Tests** tree.
Paste as Child		Paste a test element from the clipboard as a child element to the currently selected test element.
Move Up		Move a test element up within the **Tests** tree.
Move Down		Move a test element down within the **Tests** tree.
Find		Search through all test elements in the active project based on configurable parameters.
Replace		Replace instances of found values with a new value.
New Filter		Create a new custom filter for the **Tests** tree.
Filters		Lists the available filters for the **Tests** tree
Edit Filter		Edit the currently selected custom filter.
Delete Filter		Delete the currently selected custom filter.
Copy Filter		Copy the currently selected custom filter.
Show Changes		Show recent changes to the tests.
Acknowledge		Acknowledge changes to the tests.

Test Reports

This section explains the test-related reports that ship with Test Manager. Test reports give you an overview of the progress of your tests and the status of defects over a period of time or over a range of builds.

Status Reports

The following status reports are available for the **Tests** area:

Report	Description
Test Status Overview	Represents a status overview of all tests, structured by the statuses `Passed`, `Failed`, `Not Executed`, and `N/A`.
Test Status Overview (per test container)	Represents a status overview of all tests contained in a specific test container, structured by the statuses `Passed`, `Failed`, `Not Executed`, and `N/A`.
Tests per Component	Represents an overview of coverage of components by test; makes it easier to see where testing activity is needed.
Tests (per test container)	Represents a success rate for each test container by listing the number of passed tests.
Implemented Tests (per component)	Represents an overview of coverage of components by tests that have the *Implemented* attribute set to `Yes`.
Failed Tests (per component)	Represents an overview of failed tests per component; makes it easier to identify the most critical components in the environment.

Progress Reports

The following progress reports are available for the **Tests** area:

Report	Description
Test Progress Across Builds 'X' and 'Y'	Represents a trend in test progress resulting from viewing test statuses in context with builds. The user must specify a build range consisting of a start- and an end-build.
Manual Test Coverage since Build <x> (cumulative)	Represents a trend in manual test progress, excluding all kind of automated tests, resulting from viewing test statuses in context with builds. The user must specify a starting build number which is used as basis for calculating a cumulative progress.
Automated Test Coverage for Builds (non-cumulative)	Represents the automated test coverage resulting from viewing test statuses in context with builds. The user must specify a version and a product and the report shows the test coverage that could be achieved with the automated tests per build.
Test Progress this Month	Represents a trend in test progress resulting from viewing test statuses for the current month.
Specific Test Node Progress Over the Past 'X' Days	Represents a trend in requirements coverage by considering a specific test node over the past 'X' days.
Tests Created in the Past 'X' Days (per component)	Represents a listing of new tests per component over the past 'X' days. Assists in identifying components that lack testing activity.
Test Progress Over the Past 'X' Days	Represents a trend in test progress by considering test statuses over the past 'X' days.

Report	Description
Percentage Testing Success Over the Past 'X' Days (per component)	Represents a percentage listing of successful tests per component over the past 'X' days; assists in identifying the components in the environment that are most critical.

Manual Test Reports

The following manual-test reports are available for the **Tests** area:

Report	Description
Planned vs. Actual Execution Time of Manual Tests (Summary)	Represents an overview of the deviation between planned and actual time for execution of manual tests, viewed on a daily basis.
Historic Planned vs. Actual Execution Time (per user)	Represents an overview of planned and actually required execution time for completed manual tests per user over a specific period of time.
Planned vs. Actual Execution Time (status per user)	Represents progress in terms of planned vs. actual hours of currently pending manual tests per user. Manual tests are only considered if test results have been entered by the user and are assigned to the user who enters the results.
Manual Test Result Document	An easily printable manual test case result report for a single test.
Manual Test Document	An easily printable manual test case report for a single test.

Baseline Comparison Report

Reports > **Details View** > **<Active Project>** > **Test** > **Baseline Comparison** > **Baseline Comparison**

The *Baseline Comparison* report compares a baselined project with the baseline and displays the number of changed, deleted, and created tests.

Input Parameters

The input parameters for the *Baseline Comparison* report are the identifiers of the project and the baseline.

General Report Information

This section provides a description of the report and the name of the user who executed the report.

Project and Baseline Information

This section provides the information on the project and the baseline, including data related to the creation of the baseline.

Project and Baseline Summary

This section provides a sum of the created, changed, and deleted tests in the baseline and the project, in both tabular and graphical form.

Changes

This section provides a detailed list of all changed tests in the project and the baseline, with links to the tests.

Test Run Comparison Report

Test run-comparison reports present an overview of the comparison between two runs of a test.

Note: Run-comparison reports are not suitable for the comparison of manual tests to automated tests. When the name of the report includes *(Last Two Runs)*, you can compare only the last two runs of the execution plan or test.

The test run comparison report provides the following information:

- Changes to the status of the test
- Number of errors
- Number of warnings
- Context in which the test was executed
- Execution duration of the assigned tests
- Attributes and properties of the test
- Parameters of the test
- Success conditions for the test

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Test Information

This section provides the following information about the test:

Element	Description
ID	Identifier of the test.
Name	Name of the test.
Description	Description of the test.

Execution Information

This section provides the following information about each execution:

Element	Description
Execution ID	ID of each execution plan.
Execution Name	Name of each execution plan.
Run ID	ID of each execution plan run.
Product	Name of the product.
Version	Version of the product.
Build	Build of the product.

Test Run Comparison

This section identifies the following differences between the two runs:

Element	Description
Status	Status of each run.
Execution Timestamp	Timestamp of each run.
Duration	Duration of each run.
Errors	Number of errors in each test run.
Warnings	Number of warnings in each test run.
Previous Status	Status of each run previous to the last manual change.
Changed by	User who performed the last manual change to the status.
Change Comment	Describes the reason of the manual status change.

Attributes and Properties

This section identifies the attributes and properties of the two runs of the test at execution time.

Parameters

This section lists the parameters of the two runs of the test at execution time.

Success Conditions

This section lists the conditions at execution time for each of the two runs to be considered successful. If a condition is not satisfied, the test run is considered unsuccessful. Satisfied conditions are marked green, while unsatisfied conditions are marked red.

Success Conditions

Use success conditions to determine whether a test is successful or not. You can assign one or more success conditions to each test node or suite node in the **Tests** area. If a success condition is not met during the execution of the test it is assigned to, the execution run is marked as `Failed`, except for the execution time-out success condition, for which the test execution is marked as `Not Executed`. For a test package, all success conditions except the execution time-out are disabled and hidden.

In **Details View**, when you select a node in the **Tests** tree, the **Success Conditions** table in the **Properties** page of the node displays all success conditions that are configured for the selected node. The table includes the name of each condition, whether or not the condition is active, the maximal value of the condition, and whether or not the condition is inherited.

A success condition is only evaluated when it is active. To activate and deactivate success conditions see *Editing Success Conditions*. The available types of success conditions differ depending on the test type. The following success conditions are currently available in Test Manager:

Success Condition	Description
Errors Allowed	Maximal number of errors allowed for the test. This success condition is active by default.
Warnings Allowed	Maximal number of warnings allowed for the test.
Execution Time-Out [s]	Maximal time-out allowed for the test in seconds.
Page Time: Avg. Page Time [s]	Maximal allowed average time to load a page.
Page Time: Max. Page Time [s]	Maximal allowed maximum time to load a page.

Success Condition	Description
Transaction Response Time: Avg. Trans(Busy)ok [s]	Maximal allowed average response time for a transaction in the test.
Transaction Response Time: Max. Trans(Busy)ok [s]	Maximal allowed maximum response time for a transaction in the test.

The inheritance of success conditions is similar to inheritance of properties. Success conditions that are assigned to a parent node are inherited throughout all sub-folders and child tests.

Editing Success Conditions

In **Details View**, when you select a node in the **Tests** tree, the **Properties** page of the node displays the success conditions that are associated with the node. The **Success Conditions** table includes the name of each condition, whether or not the condition is active, the maximal value of the condition, and whether or not the condition is inherited.

Note: For test package nodes, all success conditions except the execution time-out are disabled and hidden.

To edit the success conditions of a test:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a test container, test folder, or test.
3. Click the **Properties** tab. The **Properties** page displays the properties of the selected tree-element.
4. Click to the right of the **Success Conditions**. The **Edit Success Conditions** dialog box displays.
5. Uncheck the **Inherit from parent** check box of any success condition you are editing.
6. Edit values as required.
7. Specify if conditions should be active or inactive by checking or un-checking their **Active** check boxes.
8. Click **OK** to save your settings.

Test Containers

This section describes how to use test containers.

Adding Test Containers

Note: We recommend to use test containers with caution, because they bind the product to the source control profile.

To add a new test container:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the root node or a container to which you want to add the new container as a sub-node.
3. Click on the toolbar or right-click on the node and choose **New Test Container**.

   You can also right-click on the node and choose **New Test Container**.

   A new container root node is added to the **Tests** tree and the **New Test Container** dialog box displays.
4. Type a **Name** and **Description** for the container.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
5. Select any pre-defined **Product** that is to be associated with this test container from the list box. For details on adding a product profile, refer to the *SilkCentral Administration Module Help*.
6. Select a pre-defined source-control profile from the **Source Control profile** list box.

If you do not intent to check out any test sources for the test container, for example if you only have manual tests, select `---`.

Note: The Test Manager execution servers retrieve the program sources for the tests included in the container from the defined source-control profile. The root node of the source control profile is set in the root node of the test container.

7. *Optional:* Check the **Clear working folder before each test execution** check box.

   If you check the check box, the source control profile working folder is cleared before a test execution is performed. For example, the sources will be checked out before each execution. This check box is unchecked by default.

   Note: If you use an external source control system, consider that using the **Clear working folder before each test execution** option in conjunction with MS VSS can lead to longer wait times than when used in conjunction with CVS or Subversion.

8. To specify the default root path where the container is to be saved, click **Browse...** and navigate to the location.
9. *Optional:* When Test Manager is integrated with SilkPerformer® 7.1 or higher, you can define the **Custom data directory** and **Custom include directory**.

   In SilkPerformer, the `Include` directory is divided into a `System Include` directory and a `Custom Include` directory. The `Data` directory is divided into a `System Data` directory and a `Custom Data` directory. For additional information, refer to the SilkPerformer documentation.
10. The **Hidden Test Properties** portion of the dialog box allows you to specify the test property types that are to be displayed on the test container's **Properties** page, and the **Properties** pages of all test folders within the container.

    These settings do not affect the display of individual tests.
11. Click **Edit** to adjust the hidden test property settings. The **Hidden Test Properties** dialog box displays.
12. Uncheck the check boxes of all test types for which you want to have properties displayed.
13. Click **OK** to save your settings and close the **Hidden Test Properties** dialog box.
14. Click **Save** to save your settings.

The new container is added to the **Tests** tree.

Adding Links to Containers

In the **Tests** tree, you can add a link to a test container in the same project. Linked test containers are displayed in read-only mode at the position in the tree where the link is inserted.

To add a link to a test container:

1. In the menu, click **Tests** > **Details View** .
2. Right-click the node in the **Tests** tree under which you want the linked test container to display.
3. Choose **New Link** if you want to link the test container at the hierarchy level of the selected node, or choose **New Child Link** to link the test container one level below the selected node. The **Select Test Container For Linking** dialog box displays.
4. Select the test container that you want to reference.

   Note: If the referenced test container and the container in which you add the link use different source-control profiles, a confirmation dialog box displays, asking you if you really want to create the link. Linking inside a test container to another test container with a different source-control profile can lead to problems when you download or execute a test within the linked container. Click **No** if you want to change the custom include directory of one of the containers first, or click **Yes** to create the link anyway.
5. Click **OK** to confirm your selection.

The linked container is placed within the selected container as a read-only entity. Any changes to the original test container will be reflected in the linked container.

Editing Test Containers

To edit the properties of a test container:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test container that you want to edit.
3. Click in the toolbar.

   You can also right-click on the test container and select **Edit**.

   The **Edit Test Container** dialog box displays.
4. Edit the properties of the test container.
5. Click **OK** to apply your changes.

Test Folders

This section describes how you can use test folders to organize your tests.

Adding Test Folders

To add a new test folder:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test element after which you want to add the new folder.
3. Click on the toolbar.

   You can also right-click on the element and choose **New Test Folder** to add the new folder after the selected element or choose **New Child Test Folder** as a sub-node to the selected element.

   Note: For test containers you can add test folders only as sub-nodes.

   A new folder node is appended to the **Tests** tree and the **New Folder** dialog box displays.
4. Provide a name and description for the folder.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
5. Click **OK** to save your settings.

The new folder is added to the **Tests** tree.

Editing Test Folders

To modify test folder properties:

1. In the menu, click **Tests** > **Details View** .
2. Select the folder in the tree.
3. Click in the toolbar.

   You can also right-click on the test folder and select **Edit**.

   The **Edit Test Folder** dialog box displays.
4. Edit the name and description of the folder as required.
5. Click **OK** to accept your changes.

Sorting Test Folders

To move a folder up or down within the **Tests** tree:

1. In the menu, click **Tests** > **Details View** .
2. Select the folder in the tree.
3. Click on the toolbar to move the folder one step up in the tree or to move the folder one step down.

Test Packages

Test packages provide support for the structure of third-party test types in Test Manager, and consist of a package root as well as an arbitrary hierarchy of suite nodes and test nodes. Test packages also provide users with detailed information about a test execution run. Test packages, suite nodes, and test nodes can be individually assigned, along with their issues and attachments, to requirements. This functionality is similar to the functionality of every other test. After a third-party test is converted into a test package, all tests contained in the package can be run individually, if the third-party test type supports conversion. Test nodes and suite nodes contained in a test package are provided with an additional property, the **External ID**.

An advantage of test packages is that the structure can be maintained automatically with every test execution. The structure of a test package can be updated according to the results of its runs. The file `<Test Manager installation folder>\wwwroot\silkroot\xsl\output.xsd` contains an XML schema for the structure of the output XML files of test packages.

Test packages enable the entire functionality of the individual tests, with the following exceptions:

- Test containers that contain test packages cannot be linked.
- Test packages cannot be data-driven because they do not possess data-driven properties.
- All success conditions except the execution time-out are disabled and hidden for test package nodes.

Note: SilkPerformer tests, SilkTest Classic tests, and manual tests cannot be converted to test packages, as the structure of these tests is supported in Test Manager by default.

The information that provides the internal structure of a test package is specific, and is no longer consistent when you copy the test package. To reuse a test package, you have to copy the test package, revert the copied package into the parent test, apply the test to your needs, and then convert the changed test to a test package.

Creating a Test Package

To create a new test package out of a third-party test:

1. Run the test once to create the `output.xml` file, which contains the structure of the test package.
   For additional information on executing a test, see *Executing Individual Tests*.
2. In the menu, click **Tests** > **Details View** .
3. In the **Tests** tree, right-click the test and choose **Convert to Test Package**. The selected test is converted to a hierarchy representing the structure of the last execution result.

Using External IDs

You can use **`External ID`**s to uniquely identify test nodes and suite nodes in test packages. An **`External ID`** is provided as a property for each test node and each suite node. The automatically generated **`External ID`** identifies a unique test method by the fully qualified name of the class and the method with an "~" prepended.

For JUnit tests, the following schema is used for the automatically generated **`External ID`**: `~<package name>.<class name>#<method name>`.

When refactoring JUnit test classes, the automatic generation of the **External ID** is not applicable, because the result information of tests previous to the refactoring will be lost when creating a new test. In this case the **External ID** for the test must be manually defined. The refactored method is re-identifiable, because the **External ID** remains unchanged while moving a JUnit test or changing its name. The **External ID** can be manually set in the source code as an annotation.

The following code example shows such an annotation for JUnit tests:

```
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;

@Retention(RetentionPolicy.RUNTIME)
public @interface ExternalId {
  String externalId();
}
```

The annotation can be used in a JUnit test to annotate classes and test methods as shown:

```
import static org.junit.Assert.*;
import org.junit.Test;

import com.borland.runner.ExternalId;

@ExternalId(externalId="JUnit4test")
public class JUnit4test {

  @Test
  @ExternalId(externalId="MyExtId1")
  public void test1() {
    ...
  }

  @Test
  @ExternalId(externalId="MyExtId2")
  public void test2() {
    ...
  }
}
```

Be aware that using **External ID**s with JUnit runner 'org.junit.runners.Parameterized' is not supported for test methods, because the **External ID** is not unique for repeated runs of a method with different parameters. As a work around an **External ID** could be specified on class level, but must be omitted on method level. An example follows:

```
@RunWith(Parameterized.class)
@ExternalId(externalId="parameterizedWithExtId")
public class TestCaseParameterizedWithExternalId {

  @Parameters
  public static Collection<Object[]> parameterFeeder() {
    return Arrays.asList(new Object[][] {
        { "param_name1", "param_value1" }, // set of parameters per run, type
matching constructor must exist!
        { "param_name3", "param_value3" },
        { "param_name2", "param_value2" },
     }
    );
  }

  private String paramName;
  private String paramValue;

  public TestCaseParameterizedWithExternalId(String paramName, String
paramValue) {
```

```
    this.paramName = paramName;
    this.paramValue = paramValue;
  }

  @Test
  public void testWithParams() {
    System.out.println(String.format("run with parameter: name='%s',
value='%s'", paramName, paramValue));
  }

}
```

Note: The setting of the **`External ID`** for a JUnit test is only possible for tests using JUnit 4.4 or higher.

Reusing a Test Package

The information that provides the internal structure of a test package is specific, and is no longer consistent when you copy the test package. To reuse a test package, you have to copy the test package, revert the copied package into the parent test, apply the test to your needs, and then convert the changed test to a test package.

To reuse the information in a test package:

1. In the menu, click **Tests** > **Details View** .
2. Right-click the test package that you want to reuse in the **Tests** tree and select **Copy**.
3. Select the node in the **Tests** tree to which you want to add the copied test package.
4. Right-click the selected node and select **Paste**, to add the copied package to the same level in the tree, or select **Paste as Child** to add the copied test package as a sub-node to the selected node.

   Caution: The information now contained in the copied test package is not consistent.
5. Right-click the copied package and select **Revert Package to Test**. The package information is removed from the parent test.
6. Edit the test to apply it to your needs.

   For additional information, see *Editing Tests*.
7. Right-click on the edited test and select **Convert to Test Package**.

You now have a copy of the original test package with different properties.

Reverting a Test Package to the Test

To revert a test package to the original test:

1. In the menu, click **Tests** > **Details View** .
2. Right-click the test package in the **Tests** tree.
3. Click **Revert Package to Test**.

The test package is reverted to the original test.

Cleaning Up a Test Package

To clean up a test package:

1. In the menu, click **Tests** > **Details View** .
2. Right-click the test package in the **Tests** tree.
3. Click **Clean Up Test Package**.

All tests that were not executed during the last execution of the test package are removed from the test package.

Test History

Test Manager provides a complete history of all changes that are made to tests. History information is read-only, and cannot be edited or permanently deleted.

The **Recent Changes** filter, which you can access by clicking on the toolbar, enables you to efficiently view and acknowledge the latest changes and additions that have been made to tests.

Note: When you delete a test element, a change entry is added to the history file of the project to which the test element belongs.

Viewing Recent Changes

To view recent changes to tests:

1. In the menu, click **Tests** > **Details View** .
2. Click to filter out all test elements except those that have been changed since your last change acknowledgement.

   Note: The recent changes filter is selected automatically in the **Filter** list box.
3. When you have reviewed the changes, you can accept them by clicking .

   The acknowledge function resets the recent changes filter.
4. Click again to remove filtering and see all tests.

Note: All test changes generate time-stamped entries in the test history.

Tracking the History of a Test

To track a test element's history:

1. In the menu, click **Tests** > **Details View** .
2. Select the test in the **Tests** tree.
3. Click the **History** tab.

The properties of the test are displayed in tabular format.

Test History Page

Tests > **Details View** > **<Test Element>** > **History**

The **History** page details the revision history of the selected test, test container, folder, or project. It also includes an entry for every project baseline that includes the selected element, with links to the baseline and the corresponding element in the baseline. If the selected element was created as part of a project baseline, the first entry in the **History** page includes links to the original project and the corresponding element in the original project. You can use the links only to access active projects or baselines. For more information on project baselines, refer to the *SilkCentral Administration Module Help*.

For manual tests, the **History** page additionally includes an entry for each version of the selected manual test, and allows you to create new versions, revert to a previous version, and view the changes between two selected versions.

For each change, the page displays the following columns:

Column	Description
Actions	Only for version entries. For a version of a manual test, the actions you can perform are **View Version**, **Revert to Version** and **Delete Version**.
Version	Only for version entries. The version of the manual test.
Comment	A comment on the changes.
Created On	Date and time of the change.
Created By	User that made the change.

Note: When the page includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the page allow you to browse through the elements included on the page one page at a time. To display all elements as a single list, click the **[All]** link.

Upload Manager

SilkCentral's Upload Manager offers a convenient means of uploading files, typically test scripts, to the SilkCentral file pool where they are accessible to Test Manager.

For information about uploading files to Issue Manager, refer to the *SilkCentral Issue Manager Help*. Through the integration of Issue Manager with TechSmith's SnagIt screen capture utility, Issue Manager users can easily capture screen images of error conditions and upload them to Issue Manager where they can attach them to existing issues or use them as the basis for new issues.

Once you have downloaded Upload Manager to your local system, you can directly start Upload Manager as a standalone application, independent of Test Manager.

You can access Upload Manager in two ways:

- Start the Upload Manager executable from the program directory to which it has been installed. For example, `C:\Program Files\Silk\SC Test Manager <version>\Upload Manager \UploadManager.exe`.
- *Issue Manager users only:* Use a hotkey keyboard combination. For example, **Ctrl+Print Screen**. This option is enabled when the SnagIt screen capture utility is configured for use with Issue Manager. For details, refer to the *SilkCentral Issue Manager Help*.

Downloading and Installing Upload Manager

To download and install Upload Manager:

1. In the menu, click **Help** > **Tools** .
2. Click the **Upload Manager** link and save `SetupUploadManager.exe` to your local system by using the download dialog box of your browser.
3. Double-click `SetupUploadManager.exe` to start the InstallShield Wizard for Upload Manager.
4. Follow the InstallShield Wizard's prompts to enter your name, the company name, and the target destination for the installation.
5. Click **Finish** to complete the installation.

Uploading Files to the SilkCentral File Pool

To upload a file to the SilkCentral file pool:

1. Start Upload Manager by double-clicking the application's executable file, `UploadManager.exe`. The **Select target location** dialog box opens.
2. Click the **SilkCentral (server file pool)** option button.

3. Click **Next**.The **Select file to upload** dialog box opens.
4. If the file you want to upload is not visible in the **Filename** field, click **Add** to browse to and select the file.

   Note: It is not possible to add file descriptions when uploading files to the file pool.
5. When the file that you want to upload displays in the **Filename** field, click **Next**. The **Specify connection parameters** dialog box opens.
6. Enter the connection parameters for your SilkCentral installation, beginning with the **Hostname** of the computer that hosts your Test Manager installation.

   The name should not include a protocol designation. For example, `10.3.2.62`.
7. Enter the installation's **Port**.

   For example, `19120`.
8. Check the **Secure** check box if the connection is a secure connection, such as HTTPS.
9. Enter the **Username** and **Password** login credentials that are required for your SilkCentral server.
10. *Optional:* Click **Set as Default** to have these parameters presented to you automatically the next time you run Upload Manager.
11. *Optional:* Click **Internet Options** to configure Internet settings for your connection, for example proxy server settings.
12. Click **Next**.

    In the **Upload files** field, verify the information you have entered. Click **Back** if you need to make any changes on a previous page.

    Accept the default setting for the **Close this window when the upload is complete** check box.
13. Click **Finish** to begin the upload process.

When the upload is complete, Upload Manager closes and the uploaded file is available in the SilkCentral file pool.

Upload Manager Command-Line Options

You can define the settings for Upload Manager through command-line options. Defining command line options optimizes SnagIt's integration with SilkCentral by automating formatting and configuration operations that you would otherwise have to execute manually with each file upload.

Note: Settings configured through command line options override any conflicting settings that may have been manually configured using the SnagIt GUI.

Use the following format to construct a command-line option:

```
UploadManager.exe {options} {UploadFileNameList}
```

- The order in which the option settings are listed does not affect the behavior of the application.
- Option parsing is case-insensitive.
- Option parameters do not require blank spaces as separators.

Upload Manager supports the following command-line options:

Option	Description
-HOSTNAME:<HostName>	The target server. For example, `-HOSTNAME:tm.mycompany.com`. **Note:** The server name is not preceded by a protocol.

Option	Description
-USERNAME:<UserName>	The user name to be used for server login. For example, `-USERNAME:admin`.
-PASSWORD:<Password>	The password to be used for server login. For example, `-PASSWORD:secret`.
-PORT:<PortNo>	The targeted server port. The default http:// port is `80`. The default https:// port is `443`. The default test server port is `19120`.For example, `-PORT:19120`.
-SECURE:<0 or 1>	For `https://myhost:myport/` connections: • `0` sets a standard HTTP connection. • `1` sets a secure HTTPS connection. For example, `-SECURE:1`.
-CONFIG: <ConfigurationId>	The configuration ID of the targeted server, used for uploading files to Issue Manager and SilkCentral: • `2` is used for uploading files to Issue Manager. • `3` is used for uploading files to the SilkCentral server file pool. For example, `CONFIG:2`.
-PROJECT:<ProjectId>	To identify the ID of an Issue Manager, move your cursor over an active project name on the Issue Manager **Projects** page. Then look for the following string in your browser's status bar: `imPrj=<project ID #>`. `0` is the default demo project in Issue Manager. For example, `-PROJECT:0`.
-DEFID:<DefectId>	The ID of a specific issue in Issue Manager to which a file is to be attached. Use `0` to create a new issue. For example, `DEFID:156`.
-DESC:<AttachmentFileDescription>	*Issue Manager only:* The description of the attached file. For example, `-DESC:Screen Shot Attachment`.
-AUTOCLOSE	Checks a check box in the Upload Manager dialog box that instructs Upload Manager to close after it successfully completes an upload.
-AUTOUPLOAD:<WizardStepIndex>	Instructs Upload Manager to attempt automatic file upload through a specified wizard step (`WizardStepIndex`). Upload Manager then prompts the user for manual input at the specified wizard step. Automatic file upload will stop at any step within which a configuration failure is detected. For example, `-AUTOUPLOAD:10`.
-VERBOSE	Causes a dialog box to display if a dependent component is missing, for example if a SilkPerformer compiler is unavailable when a SilkPerformer project is to be uploaded to SilkCentral.
<UploadFileNameList>	Use paths to upload multiple files to the SilkCentral file pool. Any parameter passed to Upload Manager that is not preceded by a dash (-) is recognized as an absolute file path. For example, `C:\TEMP\MyScreenShot.jpg`.

Tests

This section describes how to use tests in Test Manager.

Creating Tests

To create a new test:

1. In the menu, click **Tests** > **Details View** .
2. Select a container or folder node in the **Tests** tree where you want to insert a new test.
3. Click on the toolbar or right-click within the tree and choose **New Child Test**.

   A new test node is appended to the tree view, and the **New Test** dialog box opens.
4. Type a name and description for the test.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
5. Select the test type from the **Type** list box:
6. Click **Next** and proceed to the appropriate topic for the selected test type in *Configuring Test Properties*.

Note: The well-defined public API of Test Manager enables you to implement a proprietary solution that meets your automated test needs. Test Manager is open and extensible to any external tool that can be invoked from a Java implementation or through a command-line call.

Note: Throughout the test configuration process and across all test types, **Inherit from parent** check box options are provided where applicable, enabling you to accept settings of any existing parent entity.

Editing Tests

To edit a test:

1. In the menu, click **Tests** > **Details View** .
2. Select the test or the test package that you want to edit.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
3. Click **Edit** on the toolbar.

   You can also click the **Properties** tab and click **Edit** or right-click the test or test package and select **Edit**.

   The **Edit Test** dialog box displays.
4. Edit the name and description of the selected test.

   If the selected test is a test package, the **Update Package Structure on Result** check box is available. Check the **Update Package Structure on Result** check box if you want to update the structure of the test package according to the results of the test execution run.
5. Configure the properties of the test or the test package according to the test type as described in *Configuring Test Properties*.

Executing a Trial Run of a Test

You can perform a trial run of a test to check if the test works as intended.

To perform a trial run of a test:

1. In the menu, click **Tests** > **Details View** .
2. Right-click the test that you want to try out in the **Tests** tree.

3. Select **Try Run Test**.
4. The **Go To Activities** dialog box displays. Click **Yes** if you want to analyze the results on the **Activities** page, or click **No** if you want to remain on the current Web page.

   For additional information, see *Activities*.

   Note: Check the **Don't show this dialog again (during this login session)** check box if you don't want to be asked about switching to the **Activities** page again in the future. This setting will be discarded when you log out of Test Manager.

Configuring Test Properties

This section describes how to configure the test properties for each available test type.

Finding Test Properties

The **Find** command in the **Tests** area enables you to locate test property values that meet specified search criteria. Use the **Next**, **Previous**, **First**, and **Last** functions to step through the results of a search for a specified property value. **Find** is enabled across all Test Manager plug-ins and functional categories.

Note: Data-driven test property values can be found using the **Find** command.

Note: When the **Tests** tree is constrained by a filter, **Find** is only executed against those tests that are presented in the **Tests** tree after filtering.

To find a specific test property value:

1. In the menu, click **Tests** > **Details View** .
2. Click on the toolbar. The **Find** dialog box opens.
3. From the **Category** list box, select the functional category or Test Manager plug-in across which you want to search.

   Tip: When you define a custom plug-in property, it is automatically added to the list.
4. From the **Find in** list box, select the property within which the query should search for the value.

   The properties available in this list vary based on the selected category.
5. In the **Find what** text box, type the alphanumeric string the query should search for.

   Optional settings are available for qualifying the query further. Check the check boxes of those that are appropriate:
6. Choose the location in the **Tests** tree where the search should begin.

   The following options are available:

Option Button	Description
Start from selection	Specifies that the search begins from the currently selected test element.
Start from top	Specifies that the search begins from the root of the **Tests** tree.
Find in subtree only	Specifies that the search runs only in the sub-tree which has the currently selected element of the **Tests** tree as its root node.

7. Check the appropriate check boxes to further qualify the query.

   The following check boxes are available:

Check Box	Description
Case sensitive	Specifies that the string is searched case-sensitively. **Note:** When using a case sensitive SQL Server, case-insensitive searching is not supported for the following fields:

Check Box	Description
	• Test description • Manual step description • Manual step action description • Manual step expected results
Match whole word only	Specifies that search results only include complete standalone instances of the query string.
Include read-only values	Specifies that search results include text strings that cannot be directly edited because they are inherited from another test, referenced from a linked test container, or called from a data source in the course of data-driven testing.

8. Click **Find** to begin the search and advance to the first test container, test folder, or test returned by the query.
9. If your query returns multiple test elements, you are presented with options to advance through the elements.

Note: The **Find** command allows you to search test elements where the search string is an inherited value.

Replacing Test Properties

The **Replace** command enables you to replace identified property values with new values. **Replace** is enabled across all Test Manager plug-ins and functional categories.

Note: Data-driven test property values cannot be replaced using the **Replace** command.

Note: When the **Tests** tree is constrained by a filter, **Replace** is only executed against those tests that are presented in the **Tests** tree after filtering.

To replace an identified test property value:

1. In the menu, click **Tests** > **Details View** .
2. Click on the toolbar.

   The **Replace** dialog box opens.
3. From the **Category** list box, select the functional category or Test Manager plug-in across which you want to search.

   Tip: When you define a custom plug-in property, it is automatically added to the list.
4. From the **Find in** list box, select the property within which the query should search for the value.

   The properties available in this list vary based on the selected category.
5. In the **Find what** text box, type the alphanumeric string the query should search for.

   Optional settings are available for qualifying the query further. Check the check boxes of those that are appropriate:
6. In the **Replace with** text box, type the alphanumeric string with which you want to replace the found values.
7. Click **Find** to begin the search and advance to the first test container, test folder, or test returned by the query. Or click **Replace all** to replace all instances of the queried string with the replacement string.
8. If your query returns multiple test elements, you are presented with options to advance through the elements.

Note: The **Replace** command does not allow you to search test elements where the search string is an inherited value.

Configuring SilkTest Classic Plan Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure the properties of a SilkTest Classic plan test:

1. On the **New Test** dialog box, select **SilkTest Classic Plan** from the **Type** list box and then click **Next**.

   The **Plan Properties** dialog box opens.
2. In the **Plan file** text box, type the fully qualified name of the test plan file to be executed.

   Click **Browse** to browse for the file.
3. In the **SilkTest Classic project file** text box, type the name of the SilkTest Classic project containing the file and environmental settings.

   Click **Browse** to browse for the project file.
4. In the **Option set** text box, type the fully qualified name of the option set file containing environmental settings.

   Click **Browse** to browse for the option set file.
5. In the **Data file for attributes and queries** text box, type the default path of the test plan initialization file.

   Click **Browse** to browse for the test plan initialization file.
6. In the **Test plan query name** text box, type the fully qualified name of the saved test plan query.
7. Click **Finish**.

Configuring TestPartner Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure TestPartner test properties:

1. On the **New Test** dialog box, select **TestPartner Test** from the **Type** list box.
2. Click **Next**. The **Test Properties - Select Test Script** dialog box opens.
3. Click **Browse**.

   The **Choose File** dialog box opens. The assets available in this dialog box are based on the **Project Path** defined in the associated TestPartner source control profile.
4. Select a script from the list and click **OK**.

   Note: To add multiple TestPartner tests, see *Adding Multiple TestPartner Tests*.
5. In the **Playback Options** text box, type in the name of a predefined TestPartner playback option or leave the default value of `System Defaults`.
6. Click **Finish**.

Adding Multiple TestPartner Tests

To add multiple TestPartner tests, you must first follow the steps described in *Creating Tests*.

To add multiple TestPartner tests:

1. On the **New Test** dialog box, select **TestPartner Test** from the **Type** list box.
2. Click **Next**. The **Test Properties - Select Test Script** dialog box opens.
3. Click **Browse**.

   The **Choose File** dialog box opens. The assets available in this dialog box are based on the **Project Path** defined in the associated TestPartner source control profile.

4. In the **Playback Options** text box, type in the name of a predefined TestPartner playback option or leave the default value of `System Defaults`.

   Note: The import gives the defined playback option to all imported scripts. If you need to change the playback option for any scripts, edit the test after the import.

5. Click **Next**.

   The **Test Properties - Select Scripts** dialog box opens.

6. In the **TestPartner Scripts** text box, select the scripts to import by clicking `CTRL + Click`.
7. Click **Finish**.

Configuring .Net Explorer Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure .NET Explorer test properties:

1. On the **New Test** dialog box, select **.NET Explorer Test** from the **Type** list box and then click **Next**.

   The **.NET Explorer Test Properties** dialog box opens.

2. Browse to and select the .NET Explorer script to apply to the test.

   The script is a `.nef` file.

3. Browse to and select the `NetExplorer.exe` executable that executes the selected script file.

   For example `C:\Program Files\MyCustomSPFolder\DotNET Explorer\NetExplorer.exe`.

4. In the **Test case** text box, type the name of the .NET Explorer script to execute.

   If this text box is left blank, all test cases within the script are executed.

   Note: The test cases InitTestCase and EndTestCase are always executed.

5. Click **Finish**.

Configuring JUnit Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure JUnit test properties:

1. On the **New Test** dialog box, select **JUnit Test** from the **Type** list box and then click **Next**.

   The **JUnit Test Properties** dialog box opens.

2. In the **Test class** text box, type the fully qualified name of the JUnit test class.
3. In the **Test method** text box, type the name of the appropriate test method.

   The method must be available in the test class. If the **Test method** text box is left blank, all tests that are included in the suite will be executed.

4. Set the **Java home directory** to the installation path of the Java Runtime Environment (JRE).

   The path must be valid on the execution server on which the test runs.

5. Specify a valid Java **Classpath** to use on the execution server.

   We recommend to use a relative classpath. The relative classpath is then expanded to the full classpath on the execution server. By using a relative classpath, changes on the location of the source control profile do not require additional changes to the classpath.

   The relative classpath must point to the root node of the test container containing the JUnit test, for example `JUnit_tests`. The relative classpath on the execution server is then expanded to include the working folder of the source control profile, for example `C:\temp`, and the test file names, for example `JUnit4Test.jar`.

To use a previous version of JUnit, you can also add the relative classpath to `junit.jar` to the classpath, with the appropriate JUnit version. Test Manager supports JUnit 4.4 or later. For example, if the working folder of the source control profile is `C:\temp`, the specified relative paths `junit-4.4.jar;JUnit4Test.jar` are expanded to `C:\temp\JUnit_tests\junit-4.4.jar;C:\temp\JUnit_tests\JUnit4Test.jar` on the execution server. By default, if no relative classpath to the `junit.jar` is specified, Test Manager uses JUnit 4.8.1.

You can also use a fully qualified classpath. The fully qualified classpath must point to the archive or folder in which the test classes reside. Further, `junit.jar` must be added to the classpath, with the appropriate JUnit version, as the following examples show:

- `C:\Java\junit4.4\junit.jar;C:\MyApps\main.jar;C:\MyApps\utils.jar`
- `${junit_home}\junit.jar;${apps_home}\main.jar;${apps_home}\utils.jar`

6. *Optional:* In the **JVM options** text box, you can specify the command-line options and environmental variables that can affect the performance characteristics of the JVM. You can specify multiple options, but you have to type them in the right order. For example, to use the client VM and set the maximum size of the heap to 512MB, type `-client -Xmx512m`.
7. In the **Coverage path** text box, type the JAR libraries or the specific class files to monitor for code coverage information.

   We recommend using the relative coverage path from the test container root node, which is then expanded on the execution server. You can also use a fully qualified path. Use semicolons to separate multiple jar files, as the following examples show:

   - `C:\MyApps\main.jar;C:\MyApps\utils.jar`
   - `${apps_home}\main.jar;${apps_home}\utils.jar`

   Note: The coverage path setting is disregarded if the **Record external AUT Coverage** check box is checked.
8. Check the **Record external AUT Coverage** check box to get code coverage for the application under test that is defined for the execution plan in the **Code Analysis Settings** portion of **Test Manager** > **Execution Planning** > **Code Analysis** .

   If the check box is not checked, code coverage is recorded from the executing virtual machine. By default, the check box is not checked.
9. Click **Finish**.

Note: Parameters are passed to the Java process as system properties, for example `-Dhost_under_test=10.5.2.133`. Use the `System.getProperty()` method to access the system properties. For example, to access the previously passed `host_under_test property`, use `System.getProperty("host_under_test");`.

Configuring Manual Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure manual test properties:

1. On the **New Test** dialog box, select **Manual Test** from the **Type** list box.
2. In the **Planned time** text box, type the expected amount of time for this manual step to execute and then click **Next**.

   The **Add Manual Test Step** dialog box displays.

   Note: Manual test steps are automatically timed in seconds from the moment you begin execution. These values are available in **Detail** view, not **Step-by-Step** view.
3. Specify a name, an action description, and the expected results for the first step of the manual test.

Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.

4. Click **OK**.
5. *Optional:* Click **New Step** to add additional steps to your manual test.

Configuring NUnit Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

Note: We recommend that you add the `.\bin` folder of your NUnit installation to the system path. Click **Start** > **Control Panel** > **System** > **Advanced** > **Environment Variables** to add a path like `C:\Program Files\NUnit 2.2\bin` to the system environment variable PATH.

To configure NUnit test properties:

1. On the **New Test** dialog box, select **NUnit Test** from the **Type** list box and then click **Next**.

   The **NUnit Properties** dialog box displays.
2. Click **Browse** to locate and select the NUnit assembly from which you want to pull a test.
3. Type the working directory in the **NUnit Directory** text box.

   This directory is the local path to the file `nunit-console.exe`, such as `C:\Program Files\NUnit 2.2\bin`.
4. In the **NUnit Options** text box, type one or more NUnit console command-line options to specify how NUnit tests are specified.

   For example, to define that the `Flex4TestApp` NUnit test, located in `SilkTest.Ntf.Test.Flex`, should be executed, set the fixture option as follows:

   ```
   /fixture:SilkTest.Flex.Flex4TestApp
   ```

   Note: When you add multiple options, you have to separate the options by writing one option in each line in the text box.
5. Click **Finish**.

Configuring SilkPerformer Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure SilkPerformer test properties:

1. On the **New Test** dialog box, select **SilkPerformer Test** from the **Type** list box and then click **Next**.

   The **Test Properties - Select Project** dialog box opens.
2. Perform one of the following steps to define the SilkPerformer project from which your test case is taken:

   - Click **Browse** to select a SilkPerformer project that has been saved to your local file system.
   - Click **Import** to import a SilkPerformer project that is saved to the file pool. A file in the file pool can be used anywhere in the **Tests** tree.
3. On the **Import Project From File Pool** dialog box, either select a saved `.ltz` SilkPerformer project package from the **File pool entry** list box or click **Browse** to select a SilkPerformer project package that has been saved to the source-control system.

   If you check the **Remove file from file pool** check box before you click **Finish**, the selected SilkPerformer file is deleted from the file pool.
4. Perform one of the following steps to upload SilkPerformer projects to the file pool:

- From **Administration** > **Files**, click **Upload** and browse to the appropriate LTZ file. For more information, refer to the *SilkCentral Administration Module Help*.
- Use the upload mechanism offered by SilkPerformer.
- Use the Upload Manager in SilkCentral.
- Use an existing project directory by way of a UNC path. Create a new test, click **Browse** to select the appropriate LTP file, and then select a workload.

5. On the **Test Properties - Select Project** dialog box, click **Next**.
6. On the **Test Properties - Select Workload** dialog box, select one of the workload profiles that has been defined for the project from the **Workload** list box.
7. Click **Finish** to create the test case.

   Test Manager is fully integrated with SilkPerformer.

Configuring SilkTest Classic Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure SilkTest Classic test properties:

1. On the **Test Properties - Select Test Script** dialog box, click **Browse** and select the test script file from either the defined SilkTest Classic project or the source control directory.

   Express the source control directory as a relative path to the root node defined in the test container.
2. Click **Next**.

   The **Test Properties - Select Testcase** dialog box opens.

   Note: If the SilkTest Classic script is a data-driven `.g.t` file, for example `SilkTestScript1.g.t`, then data sources are completely controlled within the script file and not through the data-driven properties of Test Manager. The **Data-driven** check box is checked by default when you use a data-driven script file. For more information about data-driven SilkTest Classic tests, refer to the SilkTest Classic documentation.
3. Select a test case from the available test cases in the defined script file or specify a custom test case.
4. *Optional:* Type additional execution arguments into the **Test data** text box.

   These arguments are processed by SilkTest Classic during the execution of the test.
5. Define whether the test should provide a TrueLog.

   Note: This setting does not apply to the open agent. For more information, refer to the SilkTest Classic documentation.
6. If required, specify an option set file.
7. Click **Finish** to create the SilkTest Classic test.

   Note: If you possess SilkTest Classic test cases that require more than one hour to complete, adjust the time-out settings for Test Manager. Otherwise, Test Manager assumes an error has occurred and terminates the execution. For details about time-out settings, refer to the *SilkCentral Administration Module Help*.
8. To import multiple SilkTest Classic testcases at the same time select **SilkTest Classic Multi-testcase import** from the **Type** list box in the **New Test** dialog box and click **Next**. Follow the steps described above to complete the task.

Configuring SilkTest Workbench Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure SilkTest Workbench test properties:

1. On the **Test Properties - Select Scripts** dialog box, select one or more scripts from the **SilkTest Workbench Test** list box.
2. In the **Playback Options** text box, type in the name of a predefined SilkTest Workbench playback option or leave the default value of `System Defaults`.

   Note: A playback option must be set for a SilkTest Workbench test. The delete option on a SilkTest Workbench test will reset the playback option to the default value of `System Defaults`.
3. Click **Finish**.

Configuring Windows Scripting Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure Windows scripting test properties:

1. On the **New Test** dialog box, select **Windows Scripting Test** from the **Type** list box and then click **Next**.

   The **Windows Scripting Properties** dialog box opens.
2. Click **Browse** and select a Windows scripting test script.
3. Specify the location of any required additional parameters in the **Switches** text box.

   Note: You may add other switches to be passed to the script. For more details on the switches that can be used, see *Windows Script Host Tests* and refer to the MS Scripting Host documentation.
4. Click **Finish**.

Configuring Process Executor Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure process executor test properties:

1. On the **New Test** dialog box, select **ProcessExecutor Test** from the **Type** list box and then click **Next**. The **ProcessExecutor Test Properties** dialog box opens.
2. In the **Executable Name** text box, type the fully qualified name of the executable.
3. In the **Argument List** text box, type all arguments of the process executor test method.

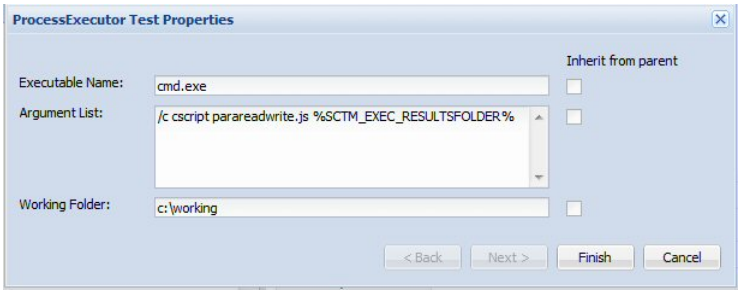
   Note: Multiple arguments must be on separate lines in the text box.
4. Set the **Working Folder**. This is the folder where the executable is executed.

During execution of the executable the following two environment variables can be used:

Environment Variable	Description
SCTM_EXEC_RESULTSFOLDER	All files in this folder are saved as result files in Test Manager. If an `output.xml` file is created during execution in this directory, it is processed by Test Manager.
SCTM_EXEC_SOURCESFOLDER	This is the folder where all source files used during the execution are located.

Example

The following example contains a ProcessExecutor test type that runs the Windows Script Host from the command line with the `/c` switch and two parameters:



- `/c` - command that specifies that the command line should terminate after execution.
- `cscript` - Windows Script Host.
- `parareadwrite.js` - the jscript file named `parareadwrite.js`
- `%SCTM_EXEC_RESULTSFOLDER%` - the Test Manager variable that contains the location of the `Result` folder on the execution server machine. Files stored in this location are automatically uploaded to Test Manager and available in the **Files** tab of the **Test Run**. In this example, the `parareadwrite.js` script produces an `output.xml` file that is written to the `Result` folder and uploaded back to Test Manager.

Configuring FitNesse Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure the properties of a FitNesse test:

1. On the **New Test** dialog box, select **FitNesse** from the **Type** list box and then click **Next**.

   The **FitNesse Test Properties** dialog box opens.
2. To specify the fully qualified URL of the FitNesse installation, choose one of the following choices:
   - Type the values of the FitNesse host, the port that is used to communicate with FitNesse, and the FitNesse test-suite into the corresponding **Host**, **Port**, and **Test Suite** text boxes. For example, type `localhost`, `8080`, and `FitNesse.SuiteAcceptanceTests`.
   - Type the fully qualified URL of the FitNesse installation into the **FitNesse URL** text box. For example, type `localhost:8080/FitNesse.SuiteAcceptanceTests`.
3. In the **Suite Filter** text box, type the name of a filter that you want to use to filter the FitNesse test-suite.
4. To specify whether FitNesse should generate the test output in HTML or XML format, type `html` or `xml` in the **Execution Method (html|xml)** text box.
5. *Optional:* In the **FitNesse URL** text box, type the URL of the computer where FitNesse is installed.

   If you leave this text box blank the test will still be executed, because the URL can be generated using the values in the **Host**, **Port** and **Test Suite** text boxes.
6. Click **Finish**.

Configuring MSTest Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

Note: The MSTest plugin supports the test type unit tests only. Results of other test types are ignored.

To configure MSTest properties:

1. On the **New Test** dialog box, select **MSTest Test** from the **Type** list.
2. Click **Next**. The **MSTest Test Properties** dialog box appears.

3. You can run a test container, a test list, a test class, or a single test method:
   - To run a test container, click **Browse** next to the **Test file** field and select a `.dll` file. You can constrain the run by typing in just a **Test class** or both a **Test class** and a **Test method**.
   - To run a test list, click **Browse** next to the **Test file** field and select a `.vsmdi` file (Visual Studio Test Meta Data). Type a name in the **Test list name** field. You can constrain the run by typing in both a **Test class** and a **Test method**.
4. Click **Finish**.

Note: To run MSTest on an execution server, a Visual Studio distribution or the Visual Studio Test Agent have to be installed. Add the folder that contains `MSTest.exe` to your path variable and restart the execution server. If Visual Studio 2010 is installed, the default path is: `C:\Program Files (x86)\Microsoft Visual Studio 10.0\Common7\IDE`. Currently MSTest distributed with Visual Studio 2010/Test Agent 2010 is supported.

Working With Tests in Grid View

The **Grid View** in the **Tests** area complements the **Document View** and **Details View** by facilitating the filtering, sorting, and grouping of large numbers of tests. **Grid View** eases identifying match points between tests and finding specific test information. Standard Windows keyboard shortcuts can be used, making it easy to select and manipulate specific sets of tests within **Grid View**. You can execute trial runs of tests and access tests from **Grid View**. You can also create execution plans through multi-selecting tests within **Grid View**.

Grid View offers a number of view-customization features that can help you better manage large numbers of tests. You can display or hide columns, adjust the width of columns, and move columns around using drag-and-drop. To enhance performance when you have numerous tests, page views are broken into views of 50 tests each. You can advance through pages using the **First**, **Last**, **Next**, and **Previous**. Or you can enter a page number into the **Page** text box and press **Enter** to access the specified page. Sorting, grouping, and filtering functions are available through context-menu commands to help you better organize your tests, group tests, and identify matching points between tests. All of your view-customization preferences are saved along with your project and will be available to you the next time you visit **Grid View**.

Accessing Tests from Grid View

You can access a test in **Tests** view directly from the **Grid View**.

To access a test's **Properties** page from **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Click on the **Name** of the test that you want to access.

   Alternatively, right click the row of the test and select **Go to test**.

Creating an Execution Plan in Grid View

To create an execution plan in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Select the tests you want to assign to your execution plan, by using the multi-select feature of the **Grid View**.
3. Right-click the tests and choose **Create Execution Plan**. The **New Execution Plan** dialog box displays.
4. Enter the specifications of your new execution plan.

Note: All selected tests must be in the same container. If not, the execution plan is not created and an error message displays. The test container is preselected in the **New Execution Plan** dialog box and can not be altered.

Modifying Properties and Attributes for Multiple Tests

To modify property or attribute values for multiple tests:

1. In the menu, click **Tests** > **Grid View** .
2. Select the tests for which you want to edit the properties or attributes.
3. Right click and select **Multi-Edit**. The **Multi-Edit** dialog box opens. This dialog lets you set the value of the selected property or attribute to the same value for all selected tests.

   Tip: Right-clicking on a cell of a specific property or attribute column pre-selects this property or attribute in the **Multi-Edit** dialog box.
4. Select **Properties** or **Attributes** from the **Category** list.
5. Select the **Name** of the property or attribute to modify.
6. Select the **Value** of the property or attribute that will apply to all selected tests.
7. Click **OK**. The selected property or attribute value is applied to all selected tests.

Displaying and Hiding Columns in Grid View

To display and hide columns in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click a column header.
3. Expand the **Columns** submenus to view all the columns that are available in the project.
4. Check the check boxes of all the columns you want to have displayed in **Grid View**. Your column-display preferences will be saved and displayed each time you open the active project.

Grouping Tests in Grid View

Beyond simply sorting by column, you can chunk tests into groups to facilitate viewing. Groups are based on commonly-shared values within the column that the grouping is based on.

To group tests in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the column that the sort is to be based on and select **Group by This Field**. The tests are organized into groups based on commonly-shared values within the column you have selected.

Removing Grouping of Tests in Grid View

To remove the grouping of tests in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click any column.
3. Uncheck the **Show in Groups** check box.

Sorting Tests in Grid View

To sort tests in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the column you want the tests to be sorted by.

3. Select **Sort Ascending** to have the tests sorted in ascending order or select **Sort Descending** to have the tests sorted in descending order. Your sort preferences are saved and displayed each time you open the active project.

Reordering Columns in Grid View

To reorder columns in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Select the column header of the column you want to move.
3. Drag the column to the desired position and release it. Your column-order preferences are saved and displayed each time you open the active project.

Resizing Columns in Grid View

To adjust the width of a column in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Select the vertical column-header divider of the column you want to adjust.
3. Drag the column boundary to the desired position and release it. Your column-width preferences are saved and displayed each time you open the active project.

Filtering Tests in Grid View

You can filter the test list based on column values. You can specify the following:

- Filter strings to be applied to text-based data fields.
- Calendar filters for date-based fields, using **Before**, **After**, or **On** operators.
- Numerical operators for number-based fields. **>**, **<**, and **=**.

*Filtering Text-Based Values in **Grid View***

To filter text-based values in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the text-based column that the filter is to be based on.
3. Expand the **Filters** submenu on the context menu to display the **Filters** text box.
4. Type the text string into the text box.
5. Press **Enter**. All tests that match the filter criteria, for example, in the case of test names, all test names that include the specified string, are then dynamically displayed in the filtered list.

*Filtering Date-Based Values in **Grid View***

To filter date-based values in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the date-based column that the filter is to be based on.
3. Hold your cursor over **Filter** on the context menu to display the **Before**, **After**, and **On** submenu.
4. Hold your cursor over **After** to define a date before which (and including) all tests should be excluded. Hold your cursor over **Before** to define a date after which (and including) all tests should be excluded. Hold your cursor over **On** to exclude all tests except those that have the specified date. The calendar tool displays.
5. Select a date using the calendar tool or click **Today** to specify today's date.

Tip: You must explicitly click a date on the calendar tool or click **Enter** to activate date-based filtering changes.

All tests that match the filter criteria are dynamically displayed in the filtered list.

*Filtering Number-Based Values in **Grid View***

To filter number-based values in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the number-based column that the filter is to be based on.
3. Expand the **Filters** submenu on the context menu to display the **>**, **<**, and **=** operators.
4. Enter a number in the **>** text box to define a number less than which (and including) all tests should be excluded. Enter a number in the **<** text box to define a number greater than which (and including) all tests should be excluded. Enter a number in the **=** text box to exclude all tests except those that have the specified number.

   Note: Number values are rounded to two decimal places.
5. Press **Enter**. All tests that match the filter criteria are dynamically displayed in the filtered list.

*Filtering Boolean Values in **Grid View***

To filter boolean values in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the boolean-based column that the filter is to be based on.
3. Expand the **Filters** submenu on the context menu to display the available values.
4. Click one of the **Yes** or **No** option buttons. All tests that match the filter criteria are dynamically displayed in the filtered list.

*Filtering Values Using a Predefined List in **Grid View***

To filter values using a predefined list in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the column that has a predefined filter value, for example `NodeType`, that the filter is to be based on.
3. Expand the **Filters** submenu on the context menu to display the available values.
4. Check the check boxes of the filter values that you are interested in. All tests having one of the selected criteria will be displayed.

*Removing Specific **Grid View** Filters*

Note: You can identify filtered columns by their titles, which are displayed in bold, italic text. Hiding a column removes all filters that have been applied to the column.

To remove a specific **Grid View** filter:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click the header of the column that has the filter you want to remove.
3. Uncheck the **Filters** check box.

*Removing All **Grid View** Filters*

Note: You can identify filtered columns by their titles, which are displayed in bold, italic text. Hiding a column removes all filters that have been applied to the column.

To remove all **Grid View** filters:

1. In the menu, click **Tests** > **Grid View** .

2. Right-click any column header and select **Reset Filters**.

Restoring Default Grid View Settings

Restoring default **Grid View** settings resets all user-defined settings for the current project. The user-defined settings are the following:

- Column order
- Column width
- Shown and hidden columns
- Applied filters
- Sorting
- Grouping

To restore the default **Grid View** settings:

1. In the menu, click **Tests** > **Grid View** .
2. Right-click any column header and select **Reset View**.

Working with Manual Tests

Steps Page

Tests > **Details View** > **<Manual Test>** > **Steps**

Tests > **Libraries of Shared Steps** > **<Shared Steps Object>** > **Steps**

The **Steps** page lists all manual test steps that are included in the selected test or shared steps object. The page supports standard Windows Explorer style multi-select functionality.

To create a new shared steps object, select the steps you want to share and click .

The page includes the following toolbar items:

Toolbar Item	Description
	Add a new test step to the end of the list.
	Insert a new test step before the selected step.
	Add a call to a shared steps object to the end of the list.
	Add a call to shared steps from a library before the selected step.
	Delete the selected test steps from the list.
	Cut the selected test step from the list and move it to the clipboard.
	Copy the selected test steps to the clipboard.
	Paste a copy of the test steps held on the clipboard to the row above the selected in the list.
	Move the selected test step one position up in the list.
	Moves the selected test step one position down in the list.
	Opens the **Attachments** dialog box, where you can perform the following actions:

Toolbar Item	Description
	Upload File Upload and attach a file to the selected test step. **Attach Link** Attach a link to the selected test step. **Delete** Delete the file or link.
	When steps of the test or shared steps object call a shared steps object in a library, click here to view the original steps. The resolved steps are marked with an arrow, and you can access them by clicking on their name. You cannot edit steps while this view is enabled.
	View steps called from a shared steps object in a library as a single step. Enable this view to edit test steps.
	Create a shared steps object out of the selected steps.
	Detach calls from other shared steps objects. Copies all steps in the calls you have selected in the grid into the manual test or shared steps object.

The **Steps** page supports the following keyboard functions (shortcuts) for test elements:

Key	Normal	SHIFT	CTRL
-			Move up
+			Move down
A			Select All
X			Cut
C			Copy
V			Paste
N			New
Ins	Insert		
Del	Delete		

The steps are displayed in a grid with the following columns:

Column	Description
Order	Number of the step in the execution sequence.
Name	Name of the test step.
Action Description	Action you must perform to execute the test step.
Expected Results	Expected result of the test step.
Attachments	Amount of links or files that are attached to the test step.

The **Edit Step** view shows the details of the selected test step and allows you to edit the name, action description, expected results, and step properties of the test step.

The following buttons are available in the **Edit Step** view:

Button	Description
OK and New Step	Click when you are finished with editing the current step and you wish to save your changes and add a new step after the current.
OK	Click when you are finished with editing the current step and you wish to save your changes.
Reset	Click to revert your changes.

Editing Manual Test Steps

Edit the manual test steps of a test or a shared steps object in the **Steps** page.

To edit a manual test step:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the node whose steps you want to edit in the **Tests** or **Libraries** tree.
3. Click the **Steps** tab.
4. Click on the test step that you want to edit. The details of the test step are shown in the **Edit Step** view.
5. Edit the name, action description, expected results, and step properties of the test step in the corresponding text boxes.

   Note: You can insert values from data sources into manual test steps in the form of parameters.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
6. Perform one of the following actions to finish editing the test step:
   - Click **Ok and New Step** to save your changes and add a new step after the current.
   - Click **Ok** to save your changes.
   - Click **Reset** to revert your changes.

Assigning Issues to Manual Test Steps

During or after an execution has run, you can assign internal or external issues to either the test itself or to the steps within the test. The most common way to do this is while running the test.

1. Click **Home** > **My Dashboard.**
2. In the **Manual Tests Assigned to Me** panel, select the test to run by clicking the **Continue Manual Test** icon. The **Run Details** screen opens.
3. Run the manual test according to its steps.
4. For test steps that have failed or are in a status that warrant an issue, in the **Test Steps** pane, select the **New Issue** or **Assign External Issue** icon in the row of the step that you want. The **New Issue** or **Assign External Issue** dialog boxes will appear.
5. Create the issue and click **OK.** After you click **OK**, the **Step Details** pane will contain a link indicating that there is an issue assigned. You can directly go to the **Issues** tab by clicking this link.

Using External Tools to Create Manual Tests

You can use the open interface of Test Manager to create manual tests outside of Test Manager's user interface. You can create your own solutions and automatically create manual tests by using Test Manager's Web Service calls.

The following calls in the `tmplanning` Web Service assist you in creating manual tests:

- getTestContainers
- addManualTest
- getCustomStepPropertyNames
- getChildNodes
- getNodeDetails
- addNode
- updateNode
- startExecution

For a detailed explanation of these Web Service calls, see *Available Web Services*.

Converting Manual Tests to Automated Tests

You can convert a manual test to one of the supported automated test types, SilkPerformer, SilkTest Classic, NUnit, and JUnit, and all installed plug-ins. The process carries manual test parameters over to the automated test, and adds automated parameters to the new automated test.

The manual test parameters that are carried over to automated tests are:

- Name
- Description
- Assigned requirements
- Assigned execution plans
- Assigned issues
- Attachments
- Test steps

Converting Manual Tests to Automated Tests

To convert a manual test to an automated test:

1. In the menu, click **Tests** > **Details View** .
2. Right-click the manual test in the **Tests** tree and select **Automate with...**.
3. Select the automated test type from the list.
4. For information on filling out the dialog, see the appropriate topic in *Configuring Test Properties*.

Printing Manual Tests

To print manual tests:

1. In the menu, click **Tests** > **Grid View** .
2. Use **Ctrl + Click** or **Shift + Click** to select the manual tests that you want to print.
3. Right-click the selection and choose **Print**. The *Manual Test Document* dialog box opens.
4. Click **Print**.

Tip: To print a single manual test, you can also right-click the test in the **Tests** tree and select **Print**.

SilkTest Classic Tests

When you observe running SilkTest Classic executions on the **Activities** page, the currently running execution plan offers a hyperlink that opens a **Details View**. This view allows you to closely monitor the state of the currently running execution plan. For SilkTest Classic test executions, the center component of this view consists of two parts: The upper part shows general information about the test, script, test case, and test data. The lower part shows all output messages generated by SilkTest Classic, along with their severity.

Test Manager's SilkTest Classic interface offers a reliable means of automating SilkTest Classic users. Each test case of a SilkTest Classic script executes within its own test execution and produces its own results.

In previous versions of Test Manager, SilkTest Classic invocation was implemented through a command-line interface. The new interface works using interprocess communication. You can specify whether or not Test Manager's SilkTest Classic interface should be used by configuring test-container settings.

For all SilkTest Classic test parameters that are not defined through SilkTest Classic test properties in the Test Manager GUI, the SilkTest Classic default settings are used, for example from `partner.ini`.

You can define the following SilkTest Classic test properties:

Test Property	Description
Test script	The test script, with file extension `.t` or `.g.t`, is defined relative to the test container's root node in the source control profile. This setting is required for all SilkTest Classic tests.
Testcase	You can select the test case from a list box or type it manually. If the test is not defined as data driven, the test case is required.
Test data	*Optional:* If several arguments are passed to SilkTest Classic, they have to be separated by a comma (,). If a `String` argument is passed to SilkTest Classic, the argument must be set in quotation marks (“). When test data is more complex, we recommend that you use parameters in the test data, for example`${ParameterName}`. Parameters are replaced automatically within test executions. **Attention:** This field has a maximum length of 2000 characters.
Data driven	When a SilkTest Classic test requires input data from an external datasource, this flag must be enabled. Default execution mode for data-driven tests is plan-based. When you use script-based execution mode for a data driven test, change the *DataDrivenScriptMode* setting in the SilkTest Classic element of `SccExecServerBootConf.xml`.
Option set	*Optional:* By default, Test Manager closes all open SilkTest Classic option set files. To specify an option set file, specify the file name relative to the test container's root node in the source control profile.

When the custom test case field is already populated, the SilkTest Classic test was automatically created, using the export functionality within SilkTest Classic. When you use the custom field to specify the test case, you can terminate the test case name with a parenthesis "()". Inside the parenthesis, you can specify test data, including parameters.

Note: Specifying data in the custom field will override the values of the *Test data* property.

Test Attributes

Attributes are administrator-created characteristics that can be applied to tests. You can filter the **Tests** area for the tests with a specific attribute and assign all such tests to execution plans. Examples include a platform attribute that can be applied to product components and a priority attribute that can be applied to tests.

Assigning Attributes to Tests

To assign an attribute to a test:

1. In the menu, click **Tests** > **Details View** .
2. Select the test to which you want to assign an attribute.
3. Click the **Attributes** tab.

4. Click **Add Attribute** to display the **Add Attributes** dialog box.
5. Click the plus symbol (+) in the **Add** column of the attribute that you want to assign. Based on the attribute type you have selected, an **Edit Attribute** dialog box displays, where you can specify which of the available attribute values you want to assign to the test.
6. Select attribute value and click **OK** to assign the attribute to the test.

Editing Test Attributes

To edit a test attribute:

1. In the menu, click **Tests** > **Details View** .
2. Select the test for which you want to edit an assigned attribute.
3. Click the **Attributes** tab.
4. Click the **Edit Attribute** button of the attribute you are editing. The **Edit Attribute** dialog box displays. The available options in the **Edit Attribute** dialog box vary depending on the attribute type that you have selected.
5. Select the required value and click **OK** to save your settings.

Deleting Attributes from Tests

To delete an attribute from a test:

1. In the menu, click **Tests** > **Details View** .
2. Select the test from which you want to delete an assigned attribute.
3. Click the **Attributes** tab.
4. Click the delete icon of the attribute you want to delete. The **Delete Attribute** confirmation dialog box displays.
5. Click **Yes** to delete the attribute.

Note: Inherited attributes cannot be deleted.

Test Attributes Page

Tests > **Details View** > **<Test Element>** > **Attributes**

The **Attributes** page in **Details View** displays all project attributes that are assigned to the selected test. For each attribute, the **Attributes** page includes the following columns:

Column	Description
Actions	Actions you can perform on the attribute. **Delete** or **Edit**.
Attribute	Name of the attribute.
Value	The value of the attribute that is assigned to the test.
Type	Attribute type.
Inherited	Whether the attribute is inherited from a parent.

Note: Inheritance of attributes is similar to inheritance of properties and success conditions. Attributes that are assigned to a parent node are inherited throughout all sub-folders and child tests.

Test Parameters

Parameters are freely configurable input values that can be assigned to different test types and used in a variety of ways. They help to define tests by defining test data.

For example, SilkPerformer tests use pre-defined parameters that represent the project attributes that are defined in a selected SilkPerformer test.

For JUnit tests, any JUnit test class can access a custom parameter of the underlying test as a Java system property; the launcher passes these parameters to the executing virtual machine using the `-D` VM argument.

For SilkTest Classic tests, parameters serve as symbols within test data properties. You can also use parameters to parameterize input data for manual test steps.

For details on custom test types, refer to the *Test Manager API Help*.

Note: For all other test types, including custom test types, only custom parameters are supported. The custom parameters are freely defined by the user. How parameters are applied for each test type varies depending on how the test types, or the plug-ins for external test types, are implemented. The built-in external test types are currently only of relevance for JUnit. The JUnit test type passes all parameters to the executing Java VM as Java system properties.

For configuration testing, you can override the test parameters in an execution plan. For additional information, see *Execution Plan Parameters*.

Creating Custom Parameters

To create a custom parameter:

1. In the menu, click **Tests** > **Details View** .
2. Select the test node for which you want to create a new parameter.
3. Click the **Parameters** tab.
4. Click **Add Custom Parameter**. The **Set Custom Parameter** dialog box displays.
5. Provide a name for the parameter.
6. In the **Type** list box, select the parameter type.
   - String
   - Number
   - Float
   - Boolean
   - Password
   - Character
7. Define the parameter value that you want to assign to the selected test.

   Note: Values for parameters of type String must be set in quotation marks (“) if you want to use the parameter in SilkTest Classic executions.
8. Click **OK**.

   Note: Parameters are automatically assigned to all sub-folders and child tests of the nodes to which they’ve been assigned.

The parameter now displays in the **Parameters** page of the selected node.

Adding Predefined Parameters to Tests

To add a predefined parameter to a test:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test node to which you want to add a predefined parameter.
3. Click the **Parameters** tab.
4. Click **Add Predefined Parameter**.

Note: For SilkPerformer tests, the **Add Predefined Parameter** button is only available for tests for which the **Project** property is already defined.

The **Add Predefined Parameter** dialog box displays, which lists all of the project attributes that are available in the project file.

5. To add any of the listed parameters, click the corresponding add icon.
6. On the dialog box that displays, specify the actual value for the parameter.
7. Click **Save** to add the parameter to the active **Tests** tree node.

Editing Predefined Parameters

To edit a predefined parameter:

1. In the menu, click **Tests** > **Details View** .
2. Select the test node for which you are editing an existing parameter.
3. Click the **Parameters** tab.
4. In the parameter you want to edit, click **Edit**. The **Set Custom Parameter** dialog box displays.
5. Edit the parameter values as required.

Note: Inherited parameters cannot be edited. Uncheck the **Inherit from parent** check box to enable editing of the parameter's **Value** setting. Parameter **Name** and **Type** settings cannot be edited.

Deleting Predefined Parameter Assignments

To delete a predefined parameter assignment:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test node for which you are deleting the assignment of an existing parameter.
3. Click the **Parameters** tab.
4. Click **Delete** in the **Actions** column of the parameter that you want to delete.

Note: Inherited parameters cannot be deleted. Uncheck the **Inherit from parent** check box on the **Set Parameter** dialog box to enable deletion of an inherited parameter.

5. Click **Yes** on the **Delete Parameter** dialog box to delete the parameter.

Parameters within Parameter Values

You can use parameter values that contain parameters. The evaluation result of such parameter values is shown in a bold font in the GUI. The following example shows how to use such a parameter:

```
parameterA := aaa
parameterB := bbb + ${parameterA}
```

Evaluated values:

```
parameterA = aaa
parameterB = bbb + aaa
```

Parameter Notations

The following parameter notations are supported:

For all tests:

```
${<parameter>}
```

All characters are allowed for parameter names, except $, {, }, and #.

Deprecated notation for manual tests:

```
#<parameter>#
```

For manual tests, the following characters are allowed for parameter names: 0-9, a-z, A-Z, and _.

Additional notation for SilkTest Classic tests:

```
$<parameter>
```

For SilkTest Classic tests, the following characters are allowed for parameter names: 0-9, a-z, A-Z, and _.

Parameter-Token Replacement Upon Execution

Any string input for a property of a test may contain placeholders in the following form: `${parametername}`. `parametername` must match the name of a parameter defined or inherited for the test. At execution time, the placeholder is replaced by the value entered for the parameter with the denoted name. This makes recurring strings in properties more customizable and facilitates the editing of common definitions.

When Test Manager finds a parameter with the notation `${<parameter>}`, it first checks if the parameter is included in the defined parameters, and if not, it checks if the parameter is an environmental variable.

For example, if the value of a JUnit classpath property is `junit.jar;${MyWorkingDir}/myclasses` and the parameter `MyWorkingDir` has the value `C:/Temp/MyWorking`, the resultant effective property value is `junit.jar;C:/Temp/MyWorking/myclasses`.

Note: The value of a parameter may also contain other parameter placeholders, which allows nesting based on the same principle.

Test Parameters Page

Tests > **Details View** > **<Test Element>** > **Parameters**

In **Details View**, the **Parameters** page displays the test parameters that are assigned to the selected test. For each parameter, the page display the following columns:

Column	Description
Actions	The actions that you can perform on the parameter. **Delete** and **Edit**.
Parameter	Name of the assigned parameter.
Value	The selected parameter value for this test.
Type	• String • Number • Float • Boolean • Password • Character
Inherited	Indicates if the parameter has been inherited from a parent.

Note: Test parameters that are contained within a property of a test, for example test-data for SilkTest Classic tests, are listed at the top of the **Parameters** page. Unused parameters are appended to the bottom of the list and grayed out, analogous to a disabled state.

Note: When the **Parameters** page includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the tab allow you to browse through the elements included on the page one page at a time. To display all elements as a single list, select the `[All]` link.

Test Parameters Page - SilkTest Workbench

Tests > **Details View** > **<Test Element>** > **Data Set**

When passing parameters from a Test Manager test to a SilkTest Workbench visual test – the following data types should be used:

Test Manager	SilkTest Workbench
String	Text
Number	Number (Long)
Number	Number (Long Long)
Number	Enumeration
Float	Number (Double)
Boolean	Boolean (True/False)

Associating Requirements with Tests

This section explains how to assign requirements to tests.

Assigning Requirements to Tests

To manually assign requirements to tests:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the test to which you want to assign requirements.
3. Click the **Assigned Requirements** tab.

   Note: If you have created a requirements filter, you can select it from the filter list above the **Available Requirements** tree. To create a new requirements filter, navigate to **Requirements** > **Details View** and click on the toolbar.

   All requirements that are available for assignment are displayed in the **Available Requirements** tree.
4. Click to the left of a requirement or double-click on the requirement to assign it to the selected test.

   Note: Newly generated tests can automatically be assigned to the requirements from which they are generated by checking the **Assign newly generated Tests to Requirements** check box on the **Generate Tests from Requirements** dialog box. This is the default behavior.

Sorting Assigned Requirements

To sort the assigned requirements:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a test.
3. Click the **Assigned Requirements** tab.
4. Click the column header of the property by which you want to sort the requirements.

   A small upward or downward pointing arrow indicates both which column the sort is based and the direction of the sort, ascending or descending.
5. If required, click the column header again to reverse the direction of the sort.

Locating Assigned Requirements

To locate assigned requirements in the **Available Requirements** tree:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a test.

3. Click the **Assigned Requirements** tab.
4. In the **Actions** column of a requirement, click to find out in which node in the **Available Requirements** tree the requirement is stored in.

The corresponding parent-requirement node is expanded and the assigned requirement is highlighted.

Removing Requirement Assignments

To remove a requirement assignment:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a test that has at least one requirement assigned to it.
3. Click the **Assigned Requirements** tab.
4. In the **Actions** column of the assigned requirement, click .
5. Click **Yes** on the confirmation dialog box to confirm deletion of the assignment.

Note: To remove all requirement assignments from the selected test, click **Remove All**.

Test Assigned Requirements Page

Tests > **Details View** > **<Test Element>** > **Assigned Requirements**

The **Assigned Requirements** page lists the requirements that have been assigned to the selected test or project, and allows you to assign additional requirements. The **Available Requirements** tree lists all requirements that can be assigned.

Note: If you have created a requirements filter, you can select it from the filter list above the **Available Requirements** tree. To create a new requirements filter, navigate to **Requirements** > **Details View** and click on the toolbar.

For each assigned requirement, the page displays the following columns:

Column	Description
Actions	Actions that you can perform on the selected requirement. **Remove Requirement**, **Locate**, and **View Description**.
Requirement	Name of the assigned requirement. Click to open the **Requirements** > **Properties** page.
Priority	Priority of the requirement.
Risk	Potential risk associated with the requirement.
Reviewed	Review status of the requirement.

Working with Test Attachments

This section describes how to work with attachments in the **Tests** area.

Attaching Files to Test Elements

To attach a file to a test element:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a container, folder, or test.
3. Click the **Attachments** tab.
4. Click **Upload File**. The **Upload File** dialog box opens.
5. Click **Browse** to select a file from your local file system.
6. Enter a description for the attachment.

7. Click **Ok** to upload the attachment to the server and associate it with the selected element.

Note: Attaching files to a test element may not work in Mozilla Firefox. Firefox requires usage of three slashes, for example `file:///`, for a file link, while other browsers require only two, for example `file://`. Additionally, Firefox includes a security feature blocking links from remote files to local files and directories. For more information, see *http://kb.mozillazine.org/Firefox_:_Issues_:_Links_to_Local_Pages_Don't_Work*.

Attaching Links to Test Elements

To attach a link to a test element:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a container, folder, or test.
3. Click the **Attachments** tab.
4. Click **Attach Link**. The **Attach URL** dialog box opens.
5. Type the URL in the **URL** text box.
6. Type a description for the attached link.
7. Click **Attach URL** to associate the link with the selected element.

Viewing Test Attachments

For a selected test element, the **Attachments** page lists the attached files and links. The attachments are by default displayed in the order in which they are uploaded, but you can sort the attachments by columns. To display the attachments that are associated with child elements of the selected element, check the **Include Child Attachments** check box.

To view a test attachment:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the element for which you want to view an attachment.
3. Click the **Attachments** tab.
4. Click on the name of the attachment.

   The name of each listed attachment serves as a link. File-attachment links open a **Save As** dialog box, enabling you to download the attachment to your local file system. Link-attachments link directly to the link destinations in a new browser window.

Deleting Attachments from Test Elements

To delete an attachment from a test element:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the element from which you want to delete an attachment.
3. Click the **Attachments** tab.
4. In the **Actions** column of the attachment you want to delete, click ✕.
5. Click **Yes** on the confirmation dialog box to delete the attachment from the project.

Note: Only one attachment at a time can be deleted.

Test Attachments Page

Tests > **Details View** > **<Test Element>** > **Attachments**

The **Attachments** page lists all files and links that have been uploaded as attachments to the selected test containers, folders, and tests. Check the **Include Child Attachments** check box to additionally display all

attachments of child tests, folders, and test containers of the selected node. For each listed attachment, the page displays the following columns:

Column	Description
Actions	Actions that can be taken on the attachment. **Delete**.
Name	Name of the attachment.
Size	Size of the attached file.
Description	Description that has been defined for the attachment.
Created On	When the attachment or link was uploaded.
Created By	User who uploaded the attachment.

Assigned Execution Plans

Viewing Assigned Executions

To view the executions that are assigned to a test:

1. In the menu, click **Tests** > **Details View** .
2. Select the test for which you want to view the assigned executions.
3. Click the **Properties** tab. The complete list of executions that are assigned to the selected test is displayed in the **Assigned Executions** grid.

Test Runs Page

Tests > **Details View** > **<Test>** > **Runs**

The **Runs** page is available on test nodes in **Details View** and offers a listing of test execution results for the selected test.

The data grid representation of the **Runs** page facilitates the filtering, sorting, and grouping of large numbers of test runs.

To compare two test runs, use **Ctrl + Shift** to select the two runs. Right click on your selection and click **Reports** > **Test Run Comparison** .

For each run, the page displays the following columns:

Column	Description	
Actions	Actions that you can perform on the test run.	
	New Issue	Click to open the **New Issue** dialog box and create a new issue for the test.
	Assign External Issue	Assign an issue from an externally-configured issue-tracking system to the test.
Run ID	The ID of the test run. Click to open the **Test Run Results** dialog box. If the test is running, click to view details of the execution.	
Run Type	The **Run Type** column shows the test type during each run. The test type might change between two runs, for example when you convert the test from manual to automated.	
Start Time	Time the run started. If the test is a manual test and currently running, Test Manager adds *(Running)* to the date and time.	
Duration	Duration of the test run in *h/mm/ss*.	

Column	Description
Execution Plan Name	The name of the assigned execution plan, or `Unassigned Tests` if the execution was a try-run or results were uploaded. Click to open the execution plan.
Status	Status of the execution. For tests that are part of a running execution plan, the status is updated in response to the current status of the test run. If the current run is aborted, the status is reset to the status before the run.
Issues Found	Displays the amount of issues that are assigned to the test run. When no issues are assigned to the test run, the column is empty. Click on the link to access the issue in the **Issues** page of **Tests** > **Details View**.
Executed By	The execution server from which the test was run.
Errors	Number of errors that were generated during the run.
Warnings	Number of warnings that were generated during the run.
Version	Version that the test was run against.
Build	Build number that the test was run against.
Execution Plan Parent	The configuration suite or folder in the context of which the execution plan is executed. Click to access the suite or folder in the **Execution Plans** tree. If the execution plan is not included in a configuration suite or folder, nothing is displayed.
Run Comment	For SilkPerformer test runs, SilkPerformer uses this column to add information to the test run when uploading results. For all other test types, you can use this column to add information to the run.

Test Issues Page

Tests > **Details View** > **<Test Element>** > **Issues**

The **Issues** page enables you to enter and track issues related to the selected test, container, or folder.

Click **New Issue** to assign a new issue to the selected test. This button is only displayed if the currently selected object is a test.

Click **Assign External Issue** to assign an issue from an external issue tracking system to the selected test. This button is only displayed if the currently selected object is a test.

For each issue, the page displays the following columns:

Column	Description
Actions	Actions that can be performed on the issue. **Delete**.
Issue ID	ID that has been automatically assigned to the issue.
Assigned Test	Test that has been assigned to the issue. This column is only displayed if the currently selected object is a container or a folder.
Synopsis	Synopsis of the issue.
Status	Status of the issue.
External ID	Indicates if the issue is tracked by an external issue tracking system. Click an external issue number to link directly to the external issue tracking system.
Test Run	The ID of the test run that the issue is assigned to. Click on the ID to access the **Details** page of the **Test Run Results** dialog box in the **Executions** area.

Column	Description
Created On	When the issue was created.
Created By	User who created the issue.
Profile	Name of the issue tracking profile, as defined in **Test Manager** > **Settings** > **Issue Tracking** .
New Issue	Assign a new issue to the selected test. This button is only displayed if the currently selected object is a test.
Assign External Issue	Assign an issue from an external issue tracking system to the selected test. This button is only displayed if the currently selected object is a test.

Data-Driven Tests

Data-driven tests are tests that are derived from values in an existing data source, such as a spreadsheet or a database. Before you can work with data-driven tests, you need to configure a data source. For details on data sources, refer to the *SilkCentral Administration Module Help*.

Creating Data-Driven Tests

To create a data-driven test:

1. In the menu, click **Tests** > **Details View** .
2. Create a new test.

   For information about creating a test, see *Creating Tests*.
3. Click the newly created test's **Properties** tab.

   Select the **Edit** icon to open the **Data-driven Properties** dialog box.
4. Scroll down to the **Data-driven Properties** section of the **Properties** tab and click . The **Data-driven Properties** dialog box opens.
5. Select a pre-configured data source from the **Data Source** list box.

   For information on configuring data sources, refer to the *SilkCentral Administration Module Help*.
6. Click **Next** to continue.
7. Select a data set from the **Data Set** list box.

   In the case of Excel data sources, this is a worksheet name. In the case of database data sources, this is a table name.
8. Check the **Each data row is a single test** check box to have each row in your data set considered to be a separate test, or do not check this check box to create a single test for all data rows of your data set.
9. *Optional:* You can enter a SQL query into the **Filter query** text box to filter your data set based on a SQL-syntax query.

   Note: Only simple `WHERE` clause queries are supported.
10. Check the **Enable data-driven properties** check box to enable data-driven functionality.
11. Click **Finish** to save your settings.

Note: Data-driven property settings are visible in the lower portion of each test's **Properties** page.

Note: To use the data-driven test functionality of Test Manager with SilkPerformer scripts, data sources with column names matching the corresponding SilkPerformer project attributes must be used in conjunction with `AttributeGet` methods.

Adding a Data Source Value to a Manual Test Step

To add a data source value to a manual test step:

1. In the menu, click **Tests** > **Details View** .
2. Create a new data-driven test.

   Select `Manual` as the test type and configure test steps.

   Note: To view the values included in your data source, click the **Data Set** tab of the test.
3. Click the **Steps** tab.
4. Select the test step that should reference the data source value.
5. In the **Action description** text box, enter a parameter that references the relevant column in your data source, using the syntax `${<column name>}`.

   For example, if you want a test step to retrieve password parameters from a spreadsheet that has a column called *Password*, you would write the parameter as `${Password}`. When you execute the manual test step, the parameter is replaced by an actual value in the corresponding data-driven data source.

Downloading CSV Data From a Data Source

To download CSV data from a data source:

1. In the menu, click **Tests** > **Details View** .
2. Select a test that relies on the data source from which you want to download data.
3. Click the **Properties** tab.
4. In the **Actions** column of either the data source or the data set, depending on which entity contains the data you want to download, click **Download**.
5. Specify the location on your local system to where the data is to be downloaded.
6. Click **OK** to download the data in CSV format.

Editing Data-Driven Properties

To edit data-driven properties:

1. In the menu, click **Tests** > **Details View** .
2. Select the test that has the property you want to edit.
3. Click the **Properties** tab.
4. In the **Actions** column of the property you are editing, click .
5. Edit the property as required.
6. Click **OK** to save your changes.

Removing Data-Driven Properties

Remove all data-driven properties when the data-driven properties are no longer relevant, and when you want to delete a data source.

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select the node from which you want to remove the data-driven properties.
3. Click the **Properties** tab.
4. In the **Actions** column of the **Data-driven Properties** section, click . All data-driven properties are removed from the node.

Note: The data-driven properties are also removed from all sub-nodes that inherit the properties of the node.

Single and Multiple Data-Driven Test Instances

When planning data-driven tests, you should first be aware of the following two different data-driven test types that are available in Test Manager:

Single data-driven test instance	A single test result is generated for all data rows of your data source. This means that the test is only successful if the execution with every single data row is successful. If the execution with one data row fails, the whole test is marked as failed.
Multiple data-driven test instance	Each data row of your data source is represented by a test of its own. This means that each data row produces a failed or passed test result. For example, if your data source is a spreadsheet with four rows, you will have the original test you created as a parent test in addition to four new child tests, one for each of the data rows.

Note: The parent test created in this process does not have parameters associated with it, since it only represents a structuring instance for its child tests and no longer functions as an actual test. All values found in the data source will be listed on the parent test's **Data Set** page.

Note: When assigning a parent test to a requirement, links to requirements are only inherited when using single data-driven test instances.

Note: You can not assign the parent test of a multiple data-driven test instance to a setup or cleanup test execution, as such a parent node is treated as a folder. You can assign one of its child nodes though, and you can also assign a single data-driven test instance to a setup or cleanup test execution.

Worksheet Handling

If your data source is a Microsoft Excel worksheet, you should follow these guidelines to ensure a successful and maintainable data-driven test setup:

- Make your column names self-describing. This will make the data source setup in Test Manager easier to maintain.
- If you use multiple worksheets, use consistent column names across the worksheets. This will make it easier for you to apply filters for selecting columns for your data source setup.
- Use certain columns as "key" columns. Key columns will allow you to maintain your data source file, while Test Manager is still able to identify specific data rows due to the value in the key column, despite changes in row orders. Values within a key column should be unique.

Data Import Considerations

When importing data rows from an external data source, Test Manager does not account for row sorting functionality used in the external data source. Due to this, the order of data rows in Test Manager might differ from the row order in the external data source. Test Manager also ignores any format settings that have been applied in the external data source. For example, if you formatted date cells in an Excel worksheet to display the date in a certain way, Test Manager will ignore this setting and import any date values in the base format `"YYYY.MM.DD HH:MM:SS.M"`.

Test Data Set Page

Tests > **Details View** > **<Test>** > **Data Set**

The **Data Set** page lists all data that is defined for data-driven testing with the selected test.

The **Filter query** row at the top of the list displays the filter value that is defined for this data set. The values of the configured data set are displayed below this row.

For each data set, the page displays the following columns:

Column	Description
Actions	Actions that can be performed on the data set or filter. **Edit** and **Delete**.
Property	The name of the data set or filter.
Value	The value of the filter or data set for the selected test.
Inherited	Indicates whether or not the data set or filter was inherited from a parent test container or test.

Note: When the page includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the page allow you to browse through the elements included on the page one page at a time. To display all elements as a single list, click the **[All]** link.

Libraries

This section provides information regarding the usage of libraries in Test Manager.

Shared Step Libraries

Shared step libraries (libraries), are collections of test steps which are shared between tests in multiple projects. Sharing the steps reduces the amount of steps that a test engineer needs to maintain.

Libraries are displayed, organized, and maintained through a hierarchical tree structure, called the **Libraries** tree. The **Libraries** tree enables you to organize shared steps in any number of hierarchy levels. You can edit or delete any node in the tree. Right click on a node in the tree to cut it, or to copy and paste it to another location. Use the **Visibility** page to define for which projects the selected library is visible. When you create a new call to a shared steps object from a project that has visibility on the library, the library is displayed in the **Call to Shared Steps** dialog box.

To view the properties of a node in the **Libraries** tree, select the node and click the **Properties** tab. To create, edit, or view step properties for a library node in the **Libraries** tree, select the node and click the **Step Properties** tab. To create, edit, or view the shared steps included in a shared steps object in the **Libraries** tree, select the shared steps object and click the **Steps** tab. The **Usages** page of each shared step object in the tree lists tests and other shared step objects that are calling the selected object, allowing you to estimate the impact of a change to the object.

When you create a baseline of a project, the baselined tests call the same shared steps objects as the original tests.

The root node of the **Libraries** tree is called **Shared Steps Libraries** and cannot be edited.

Note: Objects in the **Libraries** unit can be created, edited, and viewed by the Test Manager, Tester, and SuperUser roles. The Test Manager and SuperUser roles can additionally delete objects from the **Libraries** unit. Users with the roles Project Manager, Analyst, or Reporter can view objects in the **Libraries** unit.

Creating a Shared Step Library

1. In the menu, click **Tests** > **Libraries of Shared Steps**.
2. In the **Libraries** tree, select the root node, which is called **Shared Steps Libraries**.
3. In the toolbar, click .

   Note: You can also right click on the root node and click **New Library**.

Library Properties Page

Tests > **Libraries of Shared Steps** > **Properties**

The **Properties** page displays the following properties of the selected node in the **Libraries** tree:

Property	Description
Name	The name of the node.
ID	The identifier of the node.
Description	A description of the contents of the node.
Created On	Date and time this node was created.
Created By	Name of the user who created this node.
Changed On	The date and time this node was last changed.
Changed By	Name of the user who last changed this node.
Planned Time [hh:mm]	Estimated execution. Only for shared steps objects.

Step Properties Page

Tests > **Test Step Properties**

Tests > **Libraries of Shared Steps** > **<Library>** > **Step Properties**

The **Step Properties** page lists all properties that can be populated into manual test steps across the active project. To create a new step property, click **New Property**.

Step Property	Description
Actions	The actions that can be performed on the property are **Delete**, **Move Up**, and **Move Down**.
Name	Name of the step property.

Library Visibility Page

Tests > **Libraries of Shared Steps** > **Visibility**

The visibility of a selected library to a project is defined by the permissions of the active user. In the **Visibility** page, you can additionally enable or disable the visibility of the selected library in a specific project. To change the visibility of a the selected library to a project, click **Edit Visibility**.

When a baseline of a project that calls shared steps objects is created, the libraries that include the shared steps objects are by default visible to the new project baseline. For more information on project baselines, refer to the *SilkCentral Administration Module Help*.

For each project, the page shows the following properties:

Property	Description
Project	The name of the project.
ID	The identifier of the project.
Description	A description of the project.

Property	Description
Is Active	Whether the project is active or not.
Has Usages	Whether the project uses objects from the selected library or not.

Edit Visibility Dialog Box

The **Edit Visibility** dialog box displays the projects that can be assigned to the library by the current user.

The dialog box includes the following items:

Item	Description
All Projects	Click this option button to assign all projects to the library.
Selected Projects	Click this option button to assign selected projects from the list to the library.
Select All	Click to assign all listed projects to the library.
Deselect All	Click to unassign all listed projects from the library.

Note: Projects that are currently using the library are grayed out and you cannot unassign them from the library.

Shared Steps Objects

A shared steps object is a collection of manual test steps, included in a library, that can be used by multiple tests in multiple projects.

Calling Shared Steps

To reduce the effort for maintaining the same test steps in various tests and projects, you can add shared steps from a library as a single step to a test or shared steps object.

You can only add a call to all steps in a shared steps object, not to a selection of them. When a call to a shared steps object is added, all shared steps are displayed as a single step in the calling test or shared steps object and you can add additional steps before and after the called steps. The initial shared steps are only displayed during execution of the referencing test. The shared steps object cannot be deleted. To delete the shared steps object, you first have to remove all calls to it.

The name of the shared steps object, whose steps are called, is displayed as the name of the step in the **Steps** page. Click on the name to access the called shared steps object.

Adding a Call to Shared Steps

To add a call to a shared steps object:

1. In the menu, click **Tests** > **Details View** to add the shared steps to a test or click **Tests** > **Libraries of Shared Steps** to add the shared steps to another shared steps object.
2. In the corresponding tree, click on the node to which you want to add the shared steps.
3. Click the **Steps** tab.
4. To add the shared steps to the end of the steps list, click . To insert it above the selected step, click . The **Call to Shared Steps** dialog box opens.

5. Select the shared steps node, whose steps you want to add, from the tree.

   Note: For a test, the tree displays only the libraries that are visible to the active project. For more information, see the *Visibility Page*.
6. Click **Ok**.

Call to Shared Steps Dialog Box

The **Call to Shared Steps** dialog box enables you to browse shared steps objects in the **Libraries** tree. To open the dialog box, choose a manual test in the **Tests** tree or a shared steps object in the **Libraries** tree, click the **Steps** tab and click .

To search for existing libraries, folders, and shared steps objects in the **Libraries** tree, type the name you are searching for in the **Find** text box of the dialog box and press `Enter`. The search is case-insensitive and not limited to entire words. Libraries and folders returned by the search are displayed with their entire sub-tree.

Creating a Shared Steps Object

You can create a shared steps object in the **Steps** page of any manual test or shared steps object.

To create a shared steps object:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Navigate to the **Steps** page of the manual test or shared steps object that includes the steps you want to share.
3. Select the steps with `CTRL + CLICK` or `SHIFT + CLICK`.
4. Click . The **Create Shared Steps** dialog box opens.
5. Enter a name for the new shared steps object in the **Name** text box.
6. Choose the location where you want to place the new shared steps object in the **Libraries** tree.
7. Click **OK**. The shared steps object is created and the selected steps are replaced by a call to the shared steps object.

Detaching Shared Steps from Libraries

To make a manual test or a shared steps object independent of other shared steps objects, detach the steps from the libraries that contain the shared steps objects. All called steps are then copied into the manual test or shared steps object.

To detach the shared steps included in a manual test or a shared steps object:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. In the corresponding tree, click on the node in which you want to resolve the shared steps.
3. Click the **Steps** tab.
4. Select the calls that you want to resolve in the grid with `CTRL + CLICK` or `SHIFT + CLICK`.
5. Click and confirm the resolving in the message box.

Editing Manual Test Steps

Edit the manual test steps of a test or a shared steps object in the **Steps** page.

To edit a manual test step:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.

2. Select the node whose steps you want to edit in the **Tests** or **Libraries** tree.
3. Click the **Steps** tab.
4. Click on the test step that you want to edit. The details of the test step are shown in the **Edit Step** view.
5. Edit the name, action description, expected results, and step properties of the test step in the corresponding text boxes.

   Note: You can insert values from data sources into manual test steps in the form of parameters.

   Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
6. Perform one of the following actions to finish editing the test step:
   - Click **Ok and New Step** to save your changes and add a new step after the current.
   - Click **Ok** to save your changes.
   - Click **Reset** to revert your changes.

Steps Page

Tests > **Details View** > **<Manual Test>** > **Steps**

Tests > **Libraries of Shared Steps** > **<Shared Steps Object>** > **Steps**

The **Steps** page lists all manual test steps that are included in the selected test or shared steps object. The page supports standard Windows Explorer style multi-select functionality.

To create a new shared steps object, select the steps you want to share and click .

The page includes the following toolbar items:

Toolbar Item	Description
	Add a new test step to the end of the list.
	Insert a new test step before the selected step.
	Add a call to a shared steps object to the end of the list.
	Add a call to shared steps from a library before the selected step.
	Delete the selected test steps from the list.
	Cut the selected test step from the list and move it to the clipboard.
	Copy the selected test steps to the clipboard.
	Paste a copy of the test steps held on the clipboard to the row above the selected in the list.
	Move the selected test step one position up in the list.
	Moves the selected test step one position down in the list.
	Opens the **Attachments** dialog box, where you can perform the following actions: **Upload File** Upload and attach a file to the selected test step. **Attach Link** Attach a link to the selected test step.

Toolbar Item	Description
	Delete Delete the file or link.
	When steps of the test or shared steps object call a shared steps object in a library, click here to view the original steps. The resolved steps are marked with an arrow, and you can access them by clicking on their name. You cannot edit steps while this view is enabled.
	View steps called from a shared steps object in a library as a single step. Enable this view to edit test steps.
	Create a shared steps object out of the selected steps.
	Detach calls from other shared steps objects. Copies all steps in the calls you have selected in the grid into the manual test or shared steps object.

The **Steps** page supports the following keyboard functions (shortcuts) for test elements:

Key	Normal	SHIFT	CTRL
-			Move up
+			Move down
A			Select All
X			Cut
C			Copy
V			Paste
N			New
Ins	Insert		
Del	Delete		

The steps are displayed in a grid with the following columns:

Column	Description
Order	Number of the step in the execution sequence.
Name	Name of the test step.
Action Description	Action you must perform to execute the test step.
Expected Results	Expected result of the test step.
Attachments	Amount of links or files that are attached to the test step.

The **Edit Step** view shows the details of the selected test step and allows you to edit the name, action description, expected results, and step properties of the test step.

The following buttons are available in the **Edit Step** view:

Button	Description
OK and New Step	Click when you are finished with editing the current step and you wish to save your changes and add a new step after the current.

Button	Description
OK	Click when you are finished with editing the current step and you wish to save your changes.
Reset	Click to revert your changes.

Usages Page

Tests > **Libraries of Shared Steps** > **<Shared Steps Object>** > **Usages**

The **Usages** page displays the tests and shared steps objects that use the selected shared steps object.

To access the usages page of a shared steps object, select the shared steps object in the **Libraries** tree and click the **Usages** tab.

For each call to the selected shared steps object, the **Usages** page displays the following properties:

ID	Identifier of the calling test or shared steps object. Hidden by default.
Name	The name of the calling test or shared steps object. Click on the name to access the test or shared steps object. The icon in front of the name indicates if the calling object is a test or another shared steps object.
Created On	Date and time the calling test or shared steps object was created.
Created By	Name of the user who created the calling test or shared steps object.
Project ID	Identifier of the project in which the calling test resides. Hidden by default. Empty if the calling object is another shared steps object.
Project Name	Name of the project in which the calling test resides. Empty if the calling object is another shared steps object.

Versions

A version of a shared steps object or a manual test is a revision of the element in a certain state. The version contains the information included in the element and the individual steps of the element at the creation time of the version. Attachments, parameters, and data sets are not included in the version for manual tests.

You can manually create versions of shared steps objects or manual tests or revert to previous versions. All versions of a shared steps object or a manual test are listed in the **History** page of the element. When you copy a project, the versions are also copied.

Creating a Version

To be able to revert a set of changes you want to apply to a shared steps object or a manual test, create a version of the element.

To create a version of a shared steps object or a manual test:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the shared steps object in the **Libraries** tree, or the manual test in the **Tests** tree.
3. Click in the toolbar.

This action is only enabled if the shared steps object or the manual test was changed.

The **Create Versions** dialog box opens.

4. *Optional:* Enter a comment on the version in the **Comment** text box.
5. Click **Ok**. The new version is shown in the **History** page.

Viewing a Version

To see if a specific version of a shared steps object or a manual test applies to your testing needs, view the details of the version.

To view the details of a version of a shared steps object or a manual test:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the shared steps object in the **Libraries** tree, or the manual test in the **Tests** tree.
3. Click the **History** tab.
4. Right click on the version of which you want to see the details.
5. Click **View version <ID>**.
6. The **View Version** dialog box opens. If the version includes attachments, you can view the attachments by clicking on their number in the **Attachments** column of the **Steps** section.

Note: Use the arrows to change the version number and view additional versions.

Comparing Versions

To view the differences between two versions of a shared steps object or a manual test, compare the versions.

To compare two versions of a shared steps object or a manual test:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the shared steps object in the **Libraries** tree, or the manual test in the **Tests** tree.
3. Click the **History** tab.
4. Select the two versions you want to compare with **CTRL + CLICK**.
5. Right click on the selection.
6. Select **Compare Versions**. The **Compare Versions** dialog box opens, displaying all the added, changed, and removed content.

Note: Use the arrows to change the version numbers and compare additional versions.

Reverting to a Previous Version

To undo changes to a shared steps object or a manual test, revert to a previous version.

To revert a shared steps object or a manual test to a previous version:

1. For a shared steps object, click **Tests** > **Libraries of Shared Steps** in the menu. For a manual test, click **Tests** > **Details View** in the menu.
2. Select the shared steps object in the **Libraries** tree, or the manual test in the **Tests** tree.
3. Click the **History** tab.
4. Right-click on the version to which you want to revert the shared steps object to.
5. Click **Revert to Version <ID>**.
6. The shared steps object or manual test is reverted to the selected version and a new version is created in the **History** page. Attachments, parameters, and data sets are not reverted for manual tests.

Library History Page

The **History** page displays the history of the selected shared steps object.

The **History** page includes an entry for each version of the selected shared steps object, and allows you to create new versions, view version details, revert to a previous version, and view the differences between versions.

To display only versions in the **History** page, and no other entries, right click on the **Version** column, select **Filter** and type `0` in the **>** text box.

The page shows the following items for each listed entry:

Item	Description
Actions	Only for version entries. The actions you can perform on a selected version are **View Version**, **Delete Version** and **Revert to Version**.
Version	Only for version entries. The version of the shared steps object.
Comment	A comment on the changes.
Created On	Date and time this history entry was created.
Created By	Name of the user who created this history entry.

Windows Script Host Tests

Windows Script Host (WSH) is part of the Windows platform and creates an environment for hosting scripts. When a script is to be run at the execution server, WSH plays the role of host. It makes objects and services available for the script and provides a set of guidelines within which the script is executed. Among other things, WSH manages security and invokes the appropriate script engine.

The following online WSH resources might be of value to you:

- *http://labmice.techtarget.com/scripting/WSH.htm*
- *http://www.winguides.com/links.php?guide=scripting*
- Downloadable Script Interpreters: *http://aspn.activestate.com/ASPN/*

Supported Script Languages

WSH is language-independent for WSH-compliant scripting engines. Natively, the Windows platform supports Visual Basic Scripts, with file extension `.vbs`, and scripts written in the Java Script language, with file extension `.js`.

For other scripting languages, a dedicated script interpreter must be installed on the execution server. For example, if you install a Perl interpreter on an execution server, this will register a Perl scripting engine at the WSH environment for the extension `.pls`. Whenever a file with extension `.pls` is passed to the WSH tool, with the executable `cscript.exe`, it will invoke the appropriate interpreter because of the file extension. So the client of WSH, in this case the SilkCentral Execution Server, does not need to know about the installation of the Perl interpreter.

Note: After installing a script interpreter, for example Active Perl, try to execute a script locally on the execution server by calling the WSH command line tool with a sample script before executing the script in Test Manager. To do so, open a command shell on the execution server and type `cscript <somescript>`, where `<somescript>` is the path to a script of your choice that is available on your execution server. This is exactly what Test Manager will call when executing a WSH test on an execution server. If the script is executed, then the scripting engine has been registered successfully.

The following scripting languages are WSH compatible:

Scripting Language	File Extension
Perl	`.pls`
Python	`.py`, `.pyw`
REXX	`.REXX`
TCL	`.tcl`

WSH Test Properties

Property	Description	
Script	You can define any file where a script engine is registered for the script language the file contains. Script files under source control are deployed automatically to execution servers, comparable to test sources for other test types.	
Switches	You can enter and pass the following settings to `cscript.exe` during the execution of the test:	
	//B	Batch mode suppresses all non-command-line console UI requests from the script. We recommended that you use this option to prevent a script from waiting for user input during unattended executions at the execution server.
	//U	We recommend to that you use unicode for redirected I/O from the console.
	//T:nn	Time-out, in seconds. The maximum time the script can run, by default = no limit. This option is used to prevent excessive execution of scripts. It sets a timer. When execution time exceeds the specified value, Cscript interrupts the script engine using the `IActiveScript::InterruptThread` method and terminates the process. There is a callback hook. If the time-out is invoked, the `OnTimeOut` function is called to permit cleanup. Although it is possible to create infinite loops using this feature, it is more useful than harmful.
	//logo	Displays an execution banner at execution time that is visible at the beginning of the `log.txt` log file. This is the default setting.
	//nologo	Prevents display of the execution banner at execution time.
	//D	Enables active debugging.
	//E:engine	Use the engine to execute a script.
	//Job:xxxx	Execute a WSF job.
	//X	Execute the script in debugger.

Using Parameters in WSH Tests

Parameters that are defined for a WSH test automatically add a name and value pair to the command line as an additional argument and set the parameters as environment variables for the called process. This functionality allows you to access all parameters defined for your test within the WSH script.

For example a WSH test is defined with `myscript.js` as script and `//B` as switch. Additionally the test requires a parameter called `IPAddress` with the value `192.168.1.5` and another parameter called `Port` with the value `1492`. The resulting command line for the WSH execution in this example is:

```
csript myscript.js //B IPAddress=192.168.1.5 Port=1492
```

Viewing Information Returned from WSH

To collect results of a WSH execution, the WSH script must generate a file called `output.xml` in the current working directory of the WSH test. All files residing in this directory are stored in the database and

are downloadable through the list of files for the test execution. Files are excluded from storage when their extensions are defined under the file extensions to ignore in results property in the **Projects** area.

Note: The current working directory is dynamically created for each WSH execution. Do not use an absolute path when creating the file. Any relative path used will correctly refer to the current working directory.

Any information that a script writes to the WSH standard output goes into the `log.txt` text file that resides in the current working directory. This file is stored in the database and can be viewed as it is included in the file list of the test execution.

The following example shows how to print log information from a script:

```
WScript.Echo "This info will be written to the log.txt file"
```

The XML structure of `output.xml` begins with an element *ResultElement* that defines an attribute named *TestItem*, which specifies the name of the *ResultElement*.

The *ResultElement* must contain an element named *ErrorCount*, optionally an element named *WarningCount*, and a list of Incident elements.

The *ErrorCount* and *WarningCount* elements must contain a positive number or zero. The *ErrorCount* and *WarningCount* of the top-level *ResultElement* are used for evaluating success conditions, which determine if a test has passed or failed. The XML file might contain additional elements that are not visible in the Test Manager GUI. The `output.xml` file is however stored in the database and is viewable as it is included in the file list of the executed test.

The *Incident* element represents an event that happened during the execution of the WSH test. *Message* and *Severity* are shown in the messages list of test executions in the Test Manager GUI. An *Incident* element must contain a *Message* and a *Severity* element.

The *Severity* element must hold one of the following values:

- Info
- Warning
- Error (or Exception)
- Failure

You can store additional information in the result file. The *ResultElement* may contain any number of sub-ResultElements, so information can be easily grouped. Sub-ResultElements make the result file easier to read. For compatibility reasons related to unit tests, JUnit and NUnit, ResultElement can be named TestSuite or Test.

The *ResultElement* may contain the following additional elements:

- *FailureCount*, which is treated the same way as error count
- *RunCount*, if a test is run multiple times
- *Timer*, for example for the duration of the test
- *WasSuccess*, for compatibility with NUnit result files
- *Asserts*, for compatibility with NUnit result files

The *Incident* element may contain a list of *Detail* elements.

The *Detail* element represents detailed information about an *Incident*. It must define a *TestName* element and an *Info* element. The *TestName* is used to give detailed information about where the *Incident* happened. The *Info* element holds detailed information about the *Incident*, for example a stack trace.

Note: Up through Test Manager 8.1, the value of the *Message* and *Info* elements had to be URL encoded (ISO-8859-1). Since version 8.1.1, URL encoding is no longer allowed.

Sample Result File

```
<ResultElement TestItem="WshOutputTest">
```

```
  <ErrorCount>1</ErrorCount>
  <WarningCount>1</WarningCount>
  <Incident>
    <Message>some unexpected result</Message>
    <Severity>Error</Severity>
    <Detail>
      <TestName>function main()</TestName>
      <Info>some additional info; eg. stacktrace</Info>
    </Detail>
  </Incident>
  <Incident>
    <Message>some warning message</Message>
    <Severity>Warning</Severity>
    <Detail>
      <TestName>function main()</TestName>
      <Info>some additional info; eg. stacktrace</Info>
    </Detail>
  </Incident>
</ResultElement>
```

Java Script Sample

The following script was used to generate the sample result file. To try this script save it with the extension `.js`.

```
function dumpOutput(dumpFile)
{
  dumpFile.WriteLine("<ResultElement TestItem=\"WshOutputTest
\">");
  dumpFile.WriteLine("  <ErrorCount>1</ErrorCount>");
  dumpFile.WriteLine("  <WarningCount>1</WarningCount>");
    dumpFile.WriteLine("  <Incident>");
    dumpFile.WriteLine("    <Message>some unexpected result</
Message>");
    dumpFile.WriteLine("    <Severity>Error</Severity>");
    dumpFile.WriteLine("    <Detail>");
    dumpFile.WriteLine("      <TestName>function main()</
TestName>");
    dumpFile.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>");
    dumpFile.WriteLine("    </Detail>");
    dumpFile.WriteLine("  </Incident>");
    dumpFile.WriteLine("  <Incident>");
    dumpFile.WriteLine("    <Message>some warning message</
Message>");
    dumpFile.WriteLine("    <Severity>Warning</Severity>");
    dumpFile.WriteLine("    <Detail>");
    dumpFile.WriteLine("      <TestName>function main()</
TestName>");
    dumpFile.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>");
    dumpFile.WriteLine("    </Detail>");
    dumpFile.WriteLine("  </Incident>");
  dumpFile.WriteLine("</ResultElement>");
}

function main()
{
  var outFile;
  var fso;
  fso = WScript.CreateObject("Scripting.FileSystemObject");
  outFile = fso.CreateTextFile("output.xml", true, true);
```

```
  outFile.WriteLine("<?xml version=\"1.0\" encoding=\"UTF-16\"?
>");

  dumpOutput(outFile);
  outFile.Close();
  WScript.Echo("Test is completed");
}

main();
WScript.Quit(0);
```

Visual Basic Script Sample

The following Visual Basic script also generates the sample result file, and saves it as `Output.xml`. To try this script save it with the extension `.vbs`.

```
WScript.Echo "starting"

Dim outFile
Dim errCnt
Dim warningCnt

outFile = "output.xml"
errCnt = 1 ' retrieve that from your test results
warningCnt = 1 ' retrieve that from your test results

Set FSO = CreateObject("Scripting.FileSystemObject")
Set oTX = FSO.OpenTextFile(outFile, 2, True, -1) ' args: file,
8=append/2=overwrite, create, ASCII

oTX.WriteLine("<?xml version=""1.0"" encoding=""UTF-16""?>")
oTX.WriteLine("<ResultElement TestItem=""PerlTest"">")
oTX.WriteLine("  <ErrorCount>" & errCnt & "</ErrorCount>")
oTX.WriteLine("  <WarningCount>" & warningCnt & "</
WarningCount>")
oTX.WriteLine("  <Incident>")
oTX.WriteLine("    <Message>some unexpected result</Message>")
oTX.WriteLine("    <Severity>Error</Severity>")
oTX.WriteLine("    <Detail>")
oTX.WriteLine("      <TestName>function main()</TestName>")
oTX.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>")
oTX.WriteLine("    </Detail>")
oTX.WriteLine("  </Incident>")
oTX.WriteLine("  <Incident>")
oTX.WriteLine("    <Message>some warning message</Message>")
oTX.WriteLine("    <Severity>Warning</Severity>")
oTX.WriteLine("    <Detail>")
oTX.WriteLine("      <TestName>function main()</TestName>")
oTX.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>")
oTX.WriteLine("    </Detail>")
oTX.WriteLine("  </Incident>")
oTX.WriteLine("</ResultElement>")
```

Test Export-Update and Import

The Test Manager/Excel integration allows you two distinct options for modifying your tests. These include:

- Export-Update: this option allows you to create your tests in Test Manager, export to Excel to make the changes, and then update them back into Test Manager.
- Import: this options allows you to create tests in excel and then import them into Test Manager.

In both scenarios, you will need to a mapping file to synchronize the columns between the two products.

Note: Currently, Test Manager supports only Microsoft Excel 2007 (.xslx) files for import, and the import is limited to manual tests.

Export, Update, and Import Rules

The following business rules apply during the export-update and import processes:

General Rules

- If a test row in Excel does not have a `TestID` or a `ParentNodeID`, then it is considered to be a new test and is created in a new folder under the test container from which the import functionality was executed.
- You can add existing TestIDs to update tests on update, and you can add existing SharedStepIDs to add new shared steps on import/update.
- Data-driven tests are not supported with export-update and import functionality. Enabled data-driven tests will not be exported.
- New automated tests are not added during import and update, but existing ones are modified during an update.
- If a test is moved to a different folder hierarchy in Excel, the move is ignored during update.
- If a test is missing in Excel (it was deleted), the test is not deleted during update.
- If a test is moved from the Test Manager container after an export, no changes will occur to it during the update.
- Import uses `Parent` column for (new) folder names.
- Update uses `ParentID` column for folders for new tests (or creates a folder with date/time syntax if not specified).
- Cannot use `A-Z` as customized column name….these are reserved for Excel's `A-Z`, which can be used in mapping.
- Cannot have blank rows in Excel.

Shared steps

- Are added for import and update.
- Are deleted and reordered for update.
- Will not be updated.
- If the Excel **SharedStepID** does not exist in Test Manager, it will not be added.

Test Export and Update

This section describes how to export and to update tests from Microsoft Excel into Test Manager.

Use the export-update to Excel functionality if you want to perform bulk edits on existing tests.

Note: The following fields are not editable in Excel: **Status**, **TestType**, **StepID**, and **StepOrder**.

Export and Update Overview

The following is a high level overview for exporting and updating:

- Select the container or folder.
- Export to Excel.
- Make changes in Excel.

- Verify/create the mapping file.
- Update back into Test Manager.

Exporting a Test to Excel

1. In the menu, click **Tests** > **Details View** .
2. Right click a test container or test folder that includes the tests that you want to export.
3. Select **Export to Excel...** The **Export Test to Excel File** dialog box displays.
4. Select the mapping file from the **Choose Mapping File** list box.

   The default mapping file is `TestPropertiesDefaultMapping.xml`. For information about creating customized mapping files, see *Test Mapping File*.
5. *Optional:* Click **Edit Mapping** to edit the test mapping file.
6. Click **OK**.
7. Click **Save** on the **File Download** dialog box to save the file.

Test information is saved into the Excel file for you to edit.

Updating a Test from Excel

Before you can update a test from Excel, you first need to export the test from Test Manager.

1. In the menu, click **Tests** > **Details View** .
2. Right click a test container or folder that includes the tests that you want to update.
3. Select **Update from Excel...** The **Update Test from Excel File** dialog box opens.
4. Select the mapping file that you want to use from the **Choose Mapping File...** list.
5. To edit the mapping file, click **Edit Mapping**.
6. Click **Browse** to select the Excel file in the **Test File (.xlsx)** field.
7. Click **OK**.

Test Import

This section describes how to import tests from Microsoft Excel into Test Manager.

Test Manager enables you to import tests from Microsoft Excel (Excel) into any container or folder in the **Tests** area and then maintain and execute all included tests in Test Manager. The import is a one-time event.

Before you can import tests, you need to create a test file in Excel. For more information, see *Test File*.

Test Manager uses the information stored in the Excel file and in a mapping file to determine what data to store. Test Manager uses a mapping file to map the tests in the Excel file to the tests in the **Tests** area. A default mapping file named `TestPropertiesDefaultMapping.xml` is preselected for the import. You can choose your own customized mapping file to map your tests. For more information about mapping files, see *Test Mapping File*.

Importing a Test from an Excel File

To import a test from an Excel file into the **Tests** area:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, right-click the folder or container that you want to use as the parent node of the imported test and choose **Import from Excel File**. The **Import Test from Excel File** dialog box opens.
3. Select the mapping file from the **Choose Mapping File** list box.

   The default mapping file is `TestPropertiesDefaultMapping.xml`. For information about creating customized mapping files, see *Test Mapping File*.

4. *Optional:* Click **Edit Mapping** to edit the test mapping file.
5. Either type the fully qualified path of the test file into the **Test File** text box or click **Browse** to search for the test file.
6. Click **OK**. A message informs you if the import was successful.
7. Click **OK** to close the message.

Test File

You can create a test file in Excel to import tests into Test Manager. In the test file, you can define all tests and test steps, along with their properties, attributes, and parameters.

To map the tests in your test file to the tests in Test Manager, use an XML mapping file. The mapping file defines the location of the elements in your test file to Test Manager. You can give the items included in the test file any name you like, as long as you map them appropriately in your mapping file. For more information about mapping files, see *Test Mapping File*.

Define the following items for a test and a test step in the test file:

Item	Description
Test	Define the name of the test. Optionally, you can also define a description for the test, the estimated duration of the test, and the path to the test from the root folder. If you define no path, the test is added directly into the root folder. Additionally, you can define properties, attributes, and parameters for the test.
Test Step	Define the name of the test step. Optionally, you can also define the action that is performed in the test step, and the result that you expect of the test step. Additionally, you can define test step properties for the test step.

Note: Currently, Test Manager supports only Microsoft Excel 2007 (.xslx) files for import, and the import is limited to manual tests.

Mapping File

The mapping file is used during the update process of export-update and during the import process when data is coming into Test Manager from Excel.

Test Mapping File Contents

The XML mapping file enables Test Manager to translate the information included in the Excel test file into the actual tests. The tags in the XML file point Test Manager to the location of the test elements in the Excel test file. The default mapping file is located in `C:\Documents and Settings\All Users\Application Data\SilkCentral\OfficeImportMappingFiles\testplan\TestPropertiesDefaultMapping.xml`.

Note: The column names which are tagged in the XML mapping file can be standard Excel column names or custom names.

Administrative

The following administrative tags are required:

Tag Name	Description
HeaderRow	The row in the Excel spreadsheet in which Test Manager searches for the column headers.
FirstDataRow	The row in the Excel spreadsheet in which Test Manager seeks for the first test data. If the row is empty, Test Manager imports the next row with data.

Tag Name	Description
MultiValueSeparatorCharacters	Lists the separator characters for multi-value attributes.

Test

The following test tags are available:

Tag Name	Required	Description
Path	Yes	Identifies the column in the Excel spreadsheet that defines the path from the test tree root node to the test element.
TestName	Yes	Identifies the column in the Excel spreadsheet that defines the name of the test.
PlannedTime	No	Identifies the column in the Excel spreadsheet that defines the expected execution duration of the manual test.
TestDescription	No	Identifies the column that defines the description of the test.

Test Step Tags

If the tests include test steps, the following test-step tags are available, but not required:

Tag Name	Required	Description
StepName	Yes, if the tests include steps.	Identifies the column that defines the name of the test step.
StepActionDescription	No	Identifies the column that defines the action performed by the test step.
StepResultDescription	No	Identifies the column that defines the result of the test step.

Optional Tags

You can also define the following optional tags as necessary:

Tag Name	Description
CustomStepProperty	Any custom property that you want to define for the test step.
Attribute	Any custom attribute that you want to define for the test step. An attribute can possess multiple values. For example, `<MappingDef columnID="Level"> <Attribute multiValue="true">Level</Attribute>` defines the column named `Level` and assigns to it the multivalued attribute `Level`.
Parameter	Any custom parameter that you want to define for the test step.

Example

By default, all tags are mapped with their names, not the Excel column IDs. You can use any mix of Excel column IDs and names in your own mapping files. The following example shows a typical mapping file with Excel column IDs, such as `columnID="C"`, and names, such as `columnID="TestDefinitionName"`.

```
<HeaderRow>1</HeaderRow>
<FirstDataRow>2</FirstDataRow>
<MultiValueSeparatorCharacters>,;</MultiValueSeparatorCharacters>
<Path columnID="A"/>
<TestName columnID="TestDefinitionName"/>
<PlannedTime columnID="C"/>
```

```
<TestDescription columnID="D"/>
<StepName columnID="StepName"/>
<StepActionDescription columnID="F"/>
<StepResultDescription columnID="G"/>

<MappingDef columnID="H"><CustomStepProperty>StepProp1</CustomStepProperty></
MappingDef>
<MappingDef columnID="StepProperty2"><CustomStepProperty>StepProp2</
CustomStepProperty></MappingDef>

<MappingDef columnID="J"><Attribute>Browser</Attribute></MappingDef>
<MappingDef columnID="Level"><Attribute multiValue="true">Level</Attribute></
MappingDef>

<MappingDef columnID="Parameter1"><Parameter type="string">Para1</Parameter></
MappingDef>
<MappingDef columnID="Parameter2"><Parameter type="password">Para2</
Parameter></MappingDef>
```

Editing the Test Mapping File

To edit the test mapping file:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, right-click the folder or container that you want to use as the parent node of the imported test and choose **Import from Excel File**. The **Import Test from Excel File** dialog box opens.
3. Click **Edit Mapping**. The **Edit Mapping** dialog box opens.
4. Edit the variables of the test file in the text boxes.

   Note: For information on the variables included in the test file, see *Test File*.
5. *Optional:* Edit the name of the mapping file.
6. Click **OK** to save your changes or click **Cancel** to cancel your changes.

Test Versioning

Test Manager creates new versions of tests during the import and update processes.

When you perform an update or import from Excel into Test Manager, two new versions of the test tree are created. The versioning process will record a snapshot before and after the import/update process.

Export-Update

There are two versioning options during the export-update process:

- Tests will contain default comments to note that they were updated from excel if they are added or modified.
- You can also manually enter your own comment on the **Update Test from Excel File** dialog box during the update. This will apply for all of the test cases.

Import

Tests will contain default comments of `Created`.

Integrating Test Manager into Rally

To use the project management tool Rally with Test Manager, you have to integrate Test Manager into Rally.

A mashup is shipped with Test Manager, that enables you to display user-story related test-coverage information from Test Manager in Rally. You can link each test in Test Manager to one or more user stories

in Rally. To link tests to a user story, assign an attribute with the value of the formatted user-story ID in Rally to each test.

Creating a Mashup Tab in Rally

Create an HTML or JavaScript mashup tab in Rally to display user-story related test-coverage information.

To create a mashup tab:

1. Navigate to the tab in Rally where your mashup tab should reside.
2. Click **New Custom Tab**.
3. Type a name for your new tab in the **Name** text box. For example, `Iteration Status`.
4. *Optional:* Check the **Share with all Users** check box if you wish other users to be able to see your mashup.
5. Select **Custom Mashup** as the type of the mashup.
6. In the menu, click **Help** > **Tools** .
7. Download the **Rally Iteration Status Mashup**.
8. Paste the content of the **Rally Iteration Status Mashup** into the **HTML** text box.
9. Configure the script in the text box to match your Test Manager environment:
   - Check if the valid script source is set, for example `<script type="text/javascript" src="http://localhost:19120/silkroot/script/sctm-toolkit.js"></script>`.
   - Check if the valid URL for Test Manager is set, for example `var SCTM_URL = "http://localhost:19120/"`.
   - Check if a valid Test Manager project ID is set, for example `var SCTM_PROJECT_ID = 0`.
   - Check if a valid attribute name is set, for example `var SCTM_ATTRIBUTE_FOR_RALLY_US = "rallyattr"`.
   - Make sure usage of the `FormattedID` from Rally is enabled, `var USE_FORMATTED_ID = true`.
10. Click **Save & Close**. The mashup table is created.

Linking User Stories from Rally to Tests

Link Rally user stories to tests in Test Manager.

To link Rally user stories to Test Manager tests:

1. Select the project in Test Manager which contains the tests you want to link the Rally user stories to.
2. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
3. Click the **Attributes** tab.
4. Click **New Attribute**. The **New Attribute** dialog box opens.
5. Type `rallyattr` in the **Name** text box.
6. Select **Edit** from the **Attribute type** radio box.
7. Click **OK** to create the new attribute.
8. Assign the new attribute to each test that you want to link user stories from Rally to.
9. Set the value of the attribute to be the value of the **FormattedID** of the appropriate Rally user story for each test. If you assign multiple stories to a single test, separate the IDs with a comma and put no whitespace characters between them. For example, `US10,US12,US14`.

Execution Planning

This section explains how to manage execution plans, including assigning tests, scheduling test runs, setting up dependencies, configuring dynamic hardware-provisioning with keywords, and configuring a deployment environment.

The **Execution Planning** area enables you to maintain control over test executions during development and testing. The area enables you to configure execution plans, schedule execution plans, assign tests to execution plans, set up execution plan dependencies, configure execution-server deployment, and configure dynamic hardware-provisioning with keywords. Execution plans are displayed, organized, and maintained through a hierarchical tree structure, the **Execution Plans** tree. Each execution plan may have any number of child tests associated with it. The **Execution Plans** tree enables you to organize execution plans within folders, in any number of hierarchy levels.

Execution Planning Toolbar Functions

Execution Planning > **Details View**

Execution Planning > **Document View**

The executions toolbar provides important commands for managing your execution plans.

Note: Some commands in the execution planning toolbar are also available through context menus in the **Execution Plans** tree.

The following commands are included in the execution planning toolbar:

Command	Icon	Description
Document View		Displays the **Document View**, which shows selected properties of all executions in a single view.
Details View		Displays the **Details View**, which enables you to drill deeply into the properties of a single element of the **Execution Plans** tree.
New Child Folder		Add a new folder to the selected project or folder. This command is disabled for execution plans that are assigned to a testing cycle.
New Child Configuration Suite		Add a new configuration suite to the selected project or folder. This command is disabled for execution plans that are assigned to a testing cycle.
New Child Execution Plan		Add a new execution plan to the selected project, folder, or configuration suite. This command is disabled for execution plans that are assigned to a testing cycle.
Edit		Open the selected element of the **Execution Plans** tree for editing.
Delete		Deletes the selected element of the **Execution Plans** tree. This command is disabled for execution plans that are assigned to a testing cycle.
Cut		Cut an element of the **Execution Plans** tree and save it to the clipboard. This command is disabled for execution plans that are assigned to a testing cycle.
Copy		Copy an element of the **Execution Plans** tree to the clipboard. This command is disabled for execution plans that are assigned to a testing cycle.

Command	Icon	Description
Paste		Paste an element from the clipboard to the **Execution Plans** tree. This command is disabled for execution plans that are assigned to a testing cycle.
Paste as Child		Paste an element from the clipboard as a child to the currently selected element of the **Execution Plans** tree. This command is disabled for execution plans that are assigned to a testing cycle.
Run		Start the execution of the tests that are assigned to the execution plan. This command is disabled for execution plans that are assigned to a testing cycle.
New Filter		Create a new custom filter for the **Execution Plans** tree.
Filters		Lists the available filters for the **Execution Plans** tree
Edit Filter		Edit the currently selected custom filter.
Delete Filter		Delete the currently selected custom filter.
Copy Filter		Copy the currently selected custom filter.
Download as PDF		This function is enabled only in the **Document View**. The action generates a PDF that includes all currently visible nodes of the **Execution Plans** tree.

Manual Execution Planning

To access this page, click **Execution Planning** > **Manual Execution Planning**.

A typical challenge of planning your manual testing is that there are too many tests and not enough resources. It is important to execute the tests with the most benefit. To support you here, the **Manual Execution Planning** page is divided into two views:

Test Selection	Select manual tests and schedule them in testing cycles.
Test Assignment	Plan the execution of manual tests in detail. Determine which manual tester has to execute which tests in a certain testing cycle.

To access the **Test Selection** view, click **Execution Planning** > **Manual Execution Planning**.

To access the **Test Assignment** view, click **Test Assignment** on the top right of the **Manual Execution Planning** page. To get back to the **Test Selection** view, click **Test Selection** on the top left.

Test Selection

The **Test Selection** view is the initial view shown when you enter the **Manual Execution Planning** page. Here you can narrow down the list of manual tests with filters and quality goals, create and edit testing cycles, and assign the filtered manual tests to testing cycles.

The **Test Selection** view is divided into three areas:

- The **Filter Criteria** area on the left side. With the filters you can narrow down the tests shown in the next area.
- The **Matching Tests** area in the middle. This list shows all available tests. When you apply filters, just the filtered tests are shown.
- The **Testing Cycles** area on the right side. Here you can assign tests from the list in the middle to a testing cycle.

Filter Criteria

The **Filter Criteria** area has three boxes:

Filter Summary	Lists all filters applied to the page. Click ✕ next to a filter to remove the filter from the list.
Tests Filter	Contains a list with all test filters of the current project. Select a test filter. The filtered tests are shown in the **Matching Tests** area.
Quality Goals/ Assignment Progress	Shows a list of the quality goals in the project and the values of the quality goals. Select one or more quality goal values. The filtered tests are shown in the **Matching Tests** area. Each quality goal value has an **Assignment Progress** bar. The bar shows the number of tests that are already assigned and the number of tests that are not yet assigned to an execution plan. **Attention:** These numbers include manual and automated tests. Move the mouse over a bar to show: • The number of **Total matching tests**. • The number of **Tests to reach goal**. • The number of **Already assigned tests**. • The number of **Not yet assigned tests**. If **Already assigned tests** is greater than **Tests to reach goal** an exclamation point appears. This means that more tests are assigned than required. **Attention:** These numbers include manual and automated tests.

Note: You can only select one single test filter and one goal value per quality goal.

If you select multiple quality goal values (from different quality goals) or a quality goal value and a test filter, only those tests that match all criteria are shown in the **Matching Tests** area.

Matching Tests

The **Matching Tests** area shows the manual tests that match the selected filter criteria. By default, no quality goals or filters are selected so all manual tests display in the grid.

Select filters and quality goal values in the **Filter Criteria** area to narrow down the list of manual tests.

The caption of the **Matching Tests** grid shows:

- The number of the currently selected tests.
- The total planned time of the currently selected tests.

You can perform the following actions in the grid:

- To assign a manual test to a testing cycle, drag the test onto the header of a testing cycle.
- To assign a manual test to a manual tester, drag the test onto the tester in the testing cycle.
- Right-click on a column to sort, to group and to filter manual tests, to add or remove columns and to reset filters or the view.
- Right-click on a manual test to show the test details or to print the manual test.
- You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.

- You can multi-assign tests to different testing cycles/testers. The column **Assigned** shows a green check mark if the test is assigned at least once.
- Click to show the test details.
- Click to show the test in the **Tests Details View**.

Testing Cycles

The **Testing Cycles** area lists all testing cycles of the current project. It is the central part of the **Manual Execution Planning** page and displays on both the **Test Selection** and on the **Test Assignment** view. The testing cycles are ordered chronologically.

Click **New Testing Cycle** on the top of the **Testing Cycles** area to create a new testing cycle.

In the header of a testing cycle, click to perform the following actions:

- **Edit Testing Cycle**
- **Edit Assigned Testers**
- **Import Tests from Execution Plan**
- **Duplicate Testing Cycle**
- **Delete Testing Cycle**
- **Start Testing Cycle**
- **Finish Testing Cycle**

In a testing cycle, click on a manual tester to display the assigned tests in the **Assigned Tests** area.

Click on the header of a testing cycle to select it. The tests that are assigned to the testing cycle (but not to a specific manual tester) are listed in the **Assigned Tests** area.

If the **Assigned Tests** area is not displayed, click **Test Assignment** on the top right.

The testing cycle boxes show:

- The name of the testing cycle.
- The start and end dates.
- A burn-down chart that visualizes the testing progress. It shows how likely it is that all tests will be finished by the end of the testing cycle. Move your mouse over one of the bars to show the number of tests that are not completed.
- The percentage of the capacity that is covered. This percentage is calculated as follows: The sum of planned time of the assigned tests divided by the capacity sum of the manual testers. This percentage displays both for the testing cycle and each individual tester. If a testing cycle or a manual tester is overloaded with tests, the green box turns into red. Move the mouse over this percentage to get detailed information.
- The number of assigned tests and the completed tests for both the testing cycle and each individual tester. Move the mouse over these numbers to get detailed information.
- The manual testers that are assigned to each testing cycle. Click on the number of the assigned testers to expand or collapse the detailed information.

Creating Testing Cycles

Note: You can also create new testing cycles in the **Execution Planning Details View** and **Document View**.

To create a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the **Testing Cycles** area, click **New Testing Cycle**. The **New Testing Cycle** dialog box appears.
3. Enter a **Name** for the testing cycle.
4. Optionally enter a **Description**.

5. Click next to **Start date** to set the start date of the testing cycle.
6. Click next to **End date** to set the end date of the testing cycle.
7. In `hh:mm` format, enter the amount of time you want to provide for the testing cycle (**Capacity**).

   This capacity will be replaced by the sum of the tester capacities as soon as you add a tester to the testing cycle.
8. Click **OK**. The testing cycle is added to the **Testing Cycles** area.

Adding Manual Testers to a Testing Cycle

After you have created a testing cycle, you may want to add manual testers and capacity to it.

You can also add manual testers in the **Execution Planning Details View**. In the menu, click **Execution Planning** > **Details View**. Click a testing cycle on the left side and click **Add/Remove Testers** on the **Properties** page.

To add manual testers to a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the header of a testing cycle in the **Testing Cycles** area, click > **Edit Assigned Testers**. The **Edit Assigned Testers** dialog box appears.
3. Click **Add/Remove Testers**. The **Add/Remove Testers** dialog box appears.
4. Select the user group you want to use from the **Available** list.
5. Select one or more users from the **Available** list.

   You can select multiple users with **CTRL + CLICK** or **SHIFT + CLICK**.
6. Click **Add** or **Add All** to add users to the **Selected** list.
7. Click **OK**. The users are now added to the **Assigned Manual Tester** list.
8. Define a capacity for each user of the testing cycle in the **Capacity** field using the hh:mm format.
9. Click **Close**. The testing cycle **Capacity** is updated with the cumulative value of the capacity for each manual tester.

Editing Assigned Testers

You can add/remove manual testers to/from testing cycles and define the capacity for each individual tester.

To edit assigned testers:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the **Testing Cycles** area, in the header of a testing cycle, click > **Edit Assigned Testers**. The **Edit Assigned Testers** dialog box appears.
3. Click **Add/Remove Testers**. The **Add/Remove Testers** dialog box appears.
4. Select the user group you want to use from the **Available** list.
5. Add or remove testers:

   - Select one or more users in the **Available** list and click **Add** or **Add All**.
   - Select one or more users in the **Selected** list and click **Remove** or **Remove All**.

   You can select multiple users with **CTRL + CLICK** or **SHIFT + CLICK**.
6. Click **OK**. The users are now added to or removed from the **Assigned Testers** list.
7. Define a capacity for each user of the testing cycle in the **Capacity** field using the `hh:mm` format.

8. Click **Close**. The testing cycle **Capacity** is updated with the cumulative value of the capacity for each manual tester.

You can also edit assigned testers in the **Details View**. In the menu, click **Execution Planning** > **Details View**. Click a testing cycle on the left side and click **Add/Remove Testers** on the **Properties** page.

Assigning Tests to Testing Cycles

To assign tests to a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. Click a test in the **Matching Tests** area and drag it onto the header of a testing cycle.

   You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.

   The tests are added to the testing cycle.

Note: You can assign a test several times to different testing cycles/testers. The column **Assigned** in the **Matching Tests** area shows a green check mark if the test is assigned at least once.

Assigning Tests to Manual Testers

Note: Before you can assign tests to manual testers, you need to add manual testers to the testing cycle.

Assign tests to manual testers to distribute the workload among the testers.

You can assign a test several times to different testing cycles/testers. The column **Assigned** in the **Matching Tests** area shows a green check mark if the test is assigned at least once.

You can also assign tests to manual testers in the **Test Assignment** view. Click **Test Assignment** on the top right. Click a test in the **Assigned Tests** area and drag it onto a manual tester in a testing cycle.

To assign tests to manual testers:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In a testing cycle, click **Tester(s) assigned** to expand the list of manual testers.
3. Click a test in the **Matching Tests** area and drag it onto a manual tester in a testing cycle.

   You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.

   The tests are added to the manual tester.

Importing Tests from Execution Plan

You need the **Manage testing cycles and configurations** permission to import tests from an execution plan.

To import tests from an execution plan:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the **Testing Cycles** area, in the header of a testing cycle, click > **Import Tests from Execution Plan**. The **Select Execution Plan** dialog box appears.
3. Select an execution plan and click **OK**. All tests of the execution plan are assigned to **No specific tester** of the testing cycle.

Duplicating Testing Cycles

You need the **Manage testing cycles and configurations** permission to duplicate a testing cycle.

To duplicate a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the **Testing Cycles** area, in the header of a testing cycle, click > **Duplicate Testing Cycle**. The **Edit Testing Cycle** dialog box appears. The properties (**Name**, **Description**, **Start Date**, **End Date** and **Capacity**) are duplicated and filled in the fields.
3. Optionally adjust the properties.
4. Click **OK**. The testing cycle is duplicated with all assigned tests and testers.

Starting Testing Cycles

Note: Before starting a testing cycle, verify that the start date of the testing cycle corresponds with the current date. If not, ensure that you are able to start the testing cycle on a different date.

To start a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the **Testing Cycles** area, in the header of a testing cycle, click > **Start Testing Cycle**.

   A testing cycle can only be started once. If the testing cycle is already started, this action is disabled. A running testing cycle displays `Testing Progress` between the start and end date. A testing cycle that has not been started yet displays `Not Yet Started`. A finished testing cycle displays `Finished`.

The assigned manual tests are distributed to the manual testers. The testers are notified by email and the tests will display in their **Manual Tests Assigned to Me** panel on their dashboard.

You can still change a running test, assign additional tests, remove tests, or move tests between testers after a testing cycle has started.

Finishing Testing Cycles

Note: Before finishing a testing cycle, verify that the end date of the testing cycle corresponds with the current date. If not, ensure that you are able to finish the testing cycle on a different date.

To finish a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the **Testing Cycles** area, in the header of a testing cycle, click > **Finish Testing Cycle**.

   If the testing cycle is not yet started, this action is disabled. A testing cycle that is not running displays `Not Yet Started` between the start and end date. A running testing cycle displays `Testing Progress`. A finished testing cycle displays `Finished`.
3. The **Finish Testing Cycle** dialog box appears. Select the action you want to perform for tests that have not been executed and click **Finish**.

Note: Finished testing cycles are locked. You cannot add/remove tests or testers to/from a finished testing cycle. You can copy a test from a finished testing cycle by dragging it onto another testing cycle or tester.

Deleting Testing Cycles

Note: You can also delete testing cycles in the **Execution Planning Details View** and **Document View**. Right-click on a testing cycle and click **Delete**.

To delete a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. In the header of a testing cycle of the **Testing Cycles** area, click > **Delete Testing Cycle**.
3. Click **Yes** on the **Deletion Confirmation** dialog box.

Test Assignment

To access this page, click **Execution Planning** > **Manual Execution Planning** . On the top right, click **Test Assignment**.

The **Test Assignment** view is the second view of the **Manual Execution Planning** page. Here you can organize the manual tests that are assigned to the selected testing cycle.

Testing Cycles

The **Testing Cycles** area lists all testing cycles of the current project. It is the central part of the **Manual Execution Planning** page and displays on both the **Test Selection** and on the **Test Assignment** view. The testing cycles are ordered chronologically.

Click **New Testing Cycle** on the top of the **Testing Cycles** area to create a new testing cycle.

In the header of a testing cycle, click to perform the following actions:

- **Edit Testing Cycle**
- **Edit Assigned Testers**
- **Import Tests from Execution Plan**
- **Duplicate Testing Cycle**
- **Delete Testing Cycle**
- **Start Testing Cycle**
- **Finish Testing Cycle**

In a testing cycle, click on a manual tester to display the assigned tests in the **Assigned Tests** area.

Click on the header of a testing cycle to select it. The tests that are assigned to the testing cycle (but not to a specific manual tester) are listed in the **Assigned Tests** area.

If the **Assigned Tests** area is not displayed, click **Test Assignment** on the top right.

The testing cycle boxes show:

- The name of the testing cycle.
- The start and end dates.
- A burn-down chart that visualizes the testing progress. It shows how likely it is that all tests will be finished by the end of the testing cycle. Move your mouse over one of the bars to show the number of tests that are not completed.
- The percentage of the capacity that is covered. This percentage is calculated as follows: The sum of planned time of the assigned tests divided by the capacity sum of the manual testers. This percentage displays both for the testing cycle and each individual tester. If a testing cycle or a manual tester is overloaded with tests, the green box turns into red. Move the mouse over this percentage to get detailed information.
- The number of assigned tests and the completed tests for both the testing cycle and each individual tester. Move the mouse over these numbers to get detailed information.
- The manual testers that are assigned to each testing cycle. Click on the number of the assigned testers to expand or collapse the detailed information.

Moving Tests from One Testing Cycle to Another

You can move tests from one testing cycle to another testing cycle. That might be useful if you cannot finish all tests in time and you want to move those tests to the next testing cycle.

To move tests from one testing cycle to another testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. Click **Test Assignment** on the top right. The **Manual Execution Planning** page switches into the **Test Assignment** view.
3. In the **Testing Cycles** area on the left side, click on the header of a testing cycle.
4. In the **Assigned Tests** area, click the test you want to move and drag it onto another testing cycle.

   You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.

Attention: A finished test cannot be moved. When you move a running test to another testing cycle, the interim results are deleted. You will need to start the test again.

Note: You cannot move tests to/from a finished testing cycle. You can copy a test from a finished testing cycle by dragging it onto another testing cycle or tester.

Moving Tests from One Tester to Another

You can move tests from one manual tester to another with free capacity to distribute the workload within a testing cycle.

To move a test from one manual tester to another manual tester:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. Click **Test Assignment** on the top right. The **Manual Execution Planning** page switches into the **Test Assignment** view.
3. In the **Testing Cycles** area on the left side, click on a manual tester. The tests that are assigned to the manual tester are shown in the **Assigned Tests** area.
4. Click the test you want to move and drag it onto another manual tester.

   You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.

Attention: A finished test cannot be moved. When you move a running test to another tester, the interim results are deleted. The tester must re-start the test.

Note: You cannot move tests to/from a tester who is assigned to a finished testing cycle. You can copy a test from a finished testing cycle by dragging it onto another testing cycle or tester.

Removing Tests from Testing Cycles

To remove tests from a testing cycle:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. Click **Test Assignment** on the top right. The **Manual Execution Planning** page switches into the **Test Assignment** view.
3. In the **Testing Cycles** area on the left side, click on the header of a testing cycle. The tests that are assigned to the testing cycle are shown in the **Assigned Tests** area.
4. Click ✕ to remove a test from the testing cycle.

   You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.

Note: You cannot remove tests from a finished testing cycle. You can copy a test from a finished testing cycle by dragging it onto another testing cycle or tester.

Removing Tests from Manual Testers

To remove tests from a manual tester:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. Click **Test Assignment** on the top right. The **Manual Execution Planning** page switches into the **Test Assignment** view.
3. In the **Testing Cycles** area on the left side, click on a manual tester. The tests that are assigned to the manual tester are shown in the **Assigned Tests** area.
4. You can remove tests from the testing cycle or remove them from just the manual tester but keep them in the testing cycle:
   - To remove a test from the manual tester and the testing cycle: click .
   - To remove a test from the manual tester but keep it in the testing cycle: click a test in the **Assigned Tests** area and drag it onto **No specific tester** in the **Testing Cycle**s area.

   You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.

Note: You cannot remove tests from a manual tester who is assigned to a finished testing cycle. You can copy a test from a finished testing cycle by dragging it onto another testing cycle or tester.

Assigned Tests

The **Assigned Tests** area shows the assigned tests of the currently selected testing cycle or manual tester.

The caption of the **Assigned Tests** grid shows:

- The name of the currently selected testing cycle.
- The name of the currently selected tester.

You can perform following actions in the grid:

- To assign a manual test to a different testing cycle, drag the test onto the header of the testing cycle.
- To assign a manual test to a (different) manual tester, drag the test onto the tester in a testing cycle.
- Right-click on a column to sort, to group and to filter manual tests, to add or remove columns, and to reset filters or the view.
- Right-click on a manual test to show the test details, to specify configurations for the test, or to remove the test from the testing cycle.
- You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK.**
- Click to remove a test from the testing cycle.
- Click to show the test details.
- Click to view the manual test results.
- Click to show the test in the **Tests Details View**.
- To remove a test from a manual tester but not from the testing cycle, drag the test to **No specific tester** in the testing cycle.
- To change the execution order, click the **Order** field, type in an order number, and press `Enter`.

  Note: You should assign tests from only one test container to a testing cycle. Otherwise, order numbers are not unique. If you want to assign tests of multiple test containers, you can group the tests by the **Test Container** column.

Note: If you apply a filter or a quality goal value in the **Filter Criteria** area, the tests in the **Assigned Tests** grid that do not match the filter are disabled.

Manual Configuration Testing

You can execute a manual test with different software and hardware configurations.

To assign configurations to a manual test:

1. In the menu, click **Execution Planning** > **Manual Execution Planning**. The **Manual Execution Planning** page displays.
2. Click **Test Assignment** on the top right. The **Manual Execution Planning** page switches into the **Test Assignment** view.
3. Right-click a test in the **Assigned Tests** area.

   You can select multiple tests with **CTRL + CLICK** or **SHIFT + CLICK**.
4. Click **Specify Configurations**. The **Specify Configurations** dialog box displays.
5. Optionally click **New Configuration** and enter a name.

   To edit and organize your configurations in greater detail, click **Execution Planning** > **Configurations**.
6. Check one or more configurations. For each test/configuration combination, an instance of the test is added to the **Assigned Tests** grid.

   Note: If you uncheck a configuration, the instance for this test/configuration combination will be removed from the currently selected tester. However, instances assigned to other testers will not be removed.
7. Click **OK**.

You can assign the new test instances to different manual testers and testing cycles.

Walkthrough

This walkthrough shows you features of **Manual Execution Planning** using an example. The assumptions for the example include:

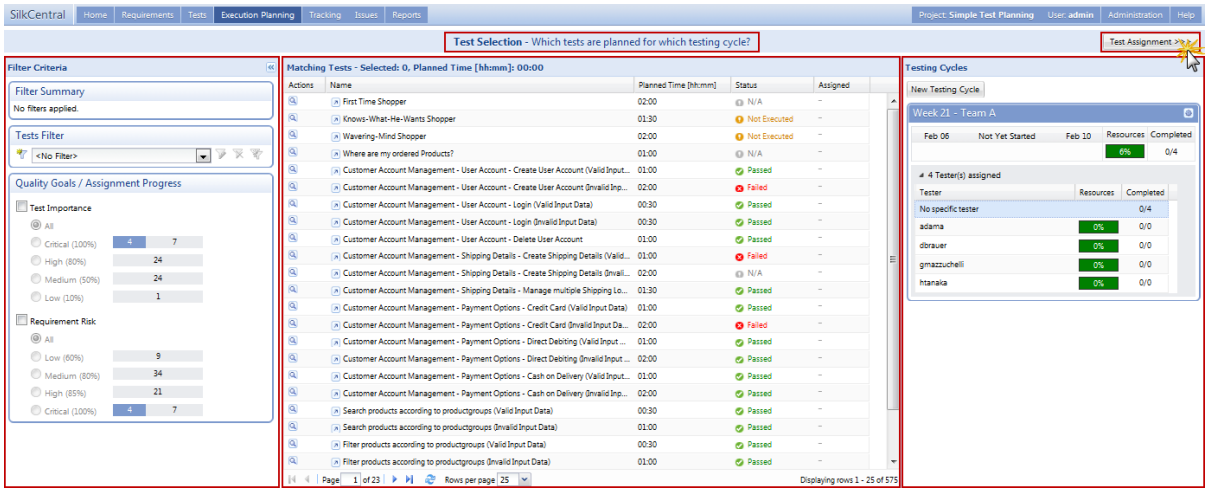
- The requirements are fully documented and categorized according to a `Risk` property.
- 575 manual tests are fully documented, assigned to requirements, and categorized according to an `Importance` attribute which defines the necessity for executing a specific test.
- Quality goals, which are also called *project exit criteria*, have been defined based on the requirements property `Risk` and the test attribute `Importance`. These quality goals specify the levels of testing required.
- The only tests to plan are the ones that best fulfill the quality goals.

Overview (Walkthrough)

The **Manual Execution Planning** page consists of four areas presented on two screens. On the first screen, the **Test Selection** screen, you can select and schedule your tests for a testing cycle. On the second screen, the **Test Assignment** screen, you can assign the tests of the selected testing cycle to the testers who will execute them.

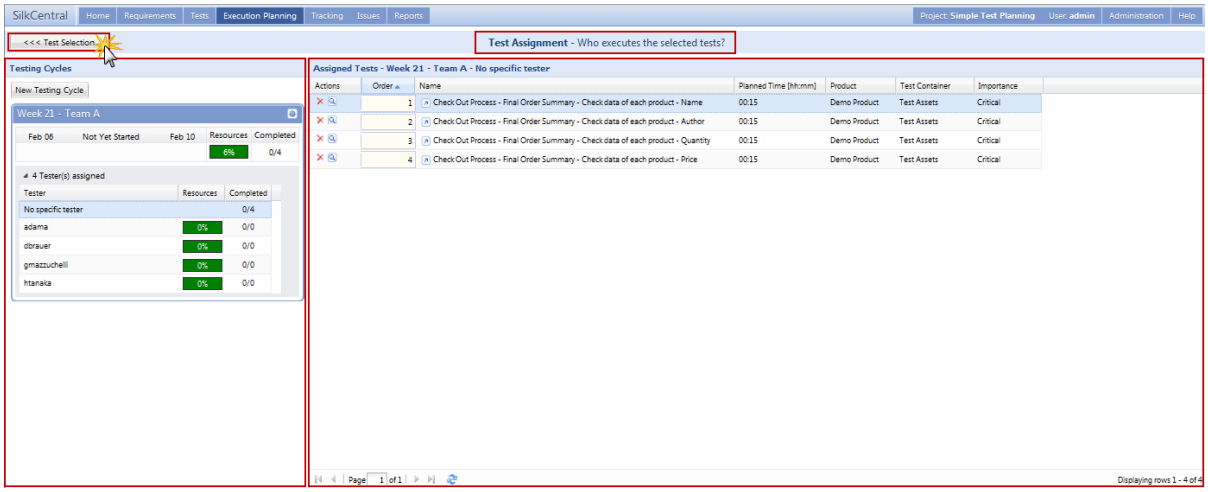
The **Test Selection** screen is divided into three areas:

- The **Filter Criteria** area on the left side. With the filter capabilities, you can narrow down the tests shown in the next area.
- The **Matching Tests** area in the middle. This list shows all available tests. When you apply filters, just the filtered tests are shown.
- The **Testing Cycles** area on the right side. Here you can assign tests from the list in the middle to a certain testing cycle, which is a defined time period.



To access the **Test Assignment** screen, click **Test Assignment >>>** on the top right. This screen is divided into two areas:

- The **Testing Cycles** area on the left side. This is the same area as on the **Test Selection** screen, it is shown on both screens.
- The **Assigned Tests** area on the right side. This list shows the tests, which have been added to the currently selected testing cycle. Here you can drag the tests from the list to the **Testing Cycles** area and drop them on a tester.



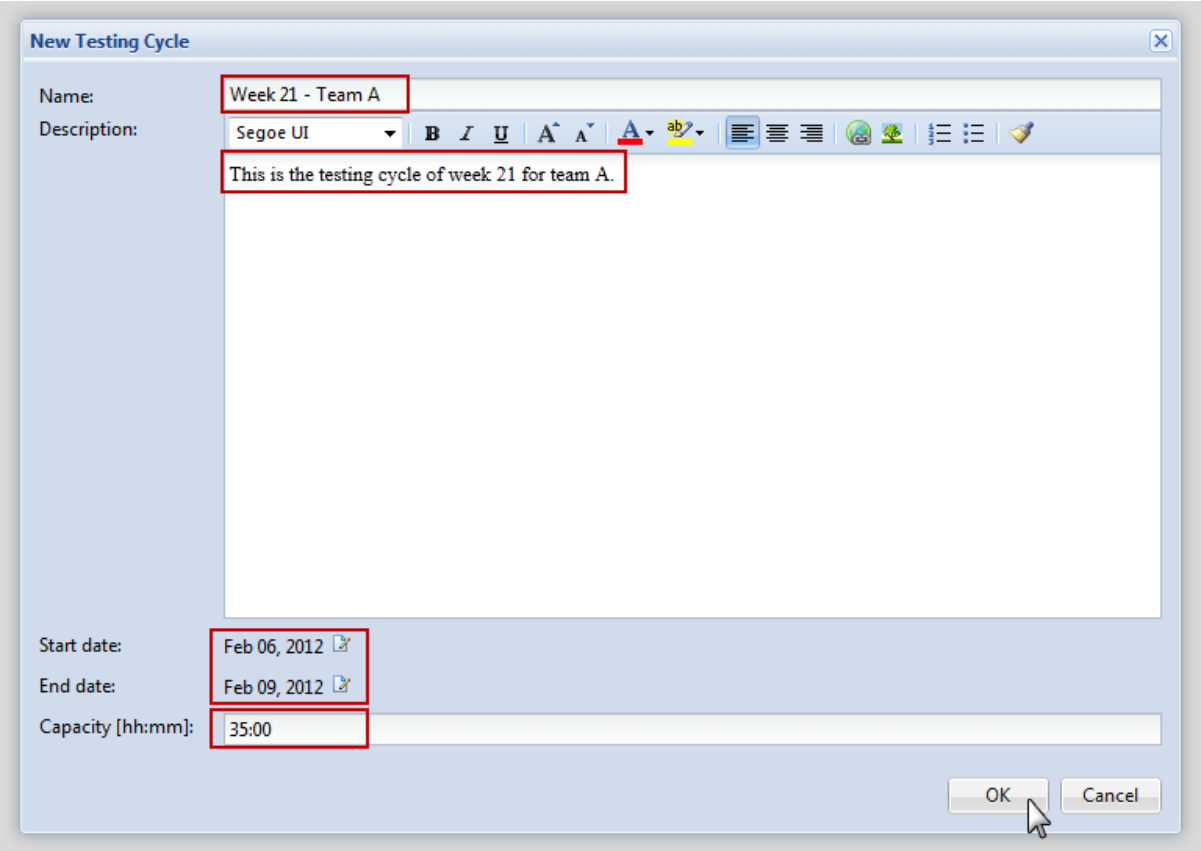
Creating a Testing Cycle (Walkthrough)

The first step is to create a testing cycle. A testing cycle is a defined period in time consisting of a start date, an end date, and a list of manual testers. We will name our testing cycle `Week 21 - Team A`.

1. In the menu, click **Execution Planning** > **Manual Execution Planning**.
2. In the **Testing Cycles** area on the right side, click **New Testing Cycle.** The **New Testing Cycle** box appears.
3. Enter a name for the testing cycle, in our case `Week 21 - Team A`.
4. Optionally enter a description.
5. Click next to **Start date** to set the start date of the testing cycle.
6. Click next to **End date** to set the end date of the testing cycle.
7. In `hh:mm` format, enter the amount of time that you want to provide for this testing cycle (**Capacity**), in our case, we entered `35:00`.

8. Click **OK**.

Our testing cycle now appears in the **Testing Cycles** area. It shows that currently no testers are assigned. The next step is to assign testers.

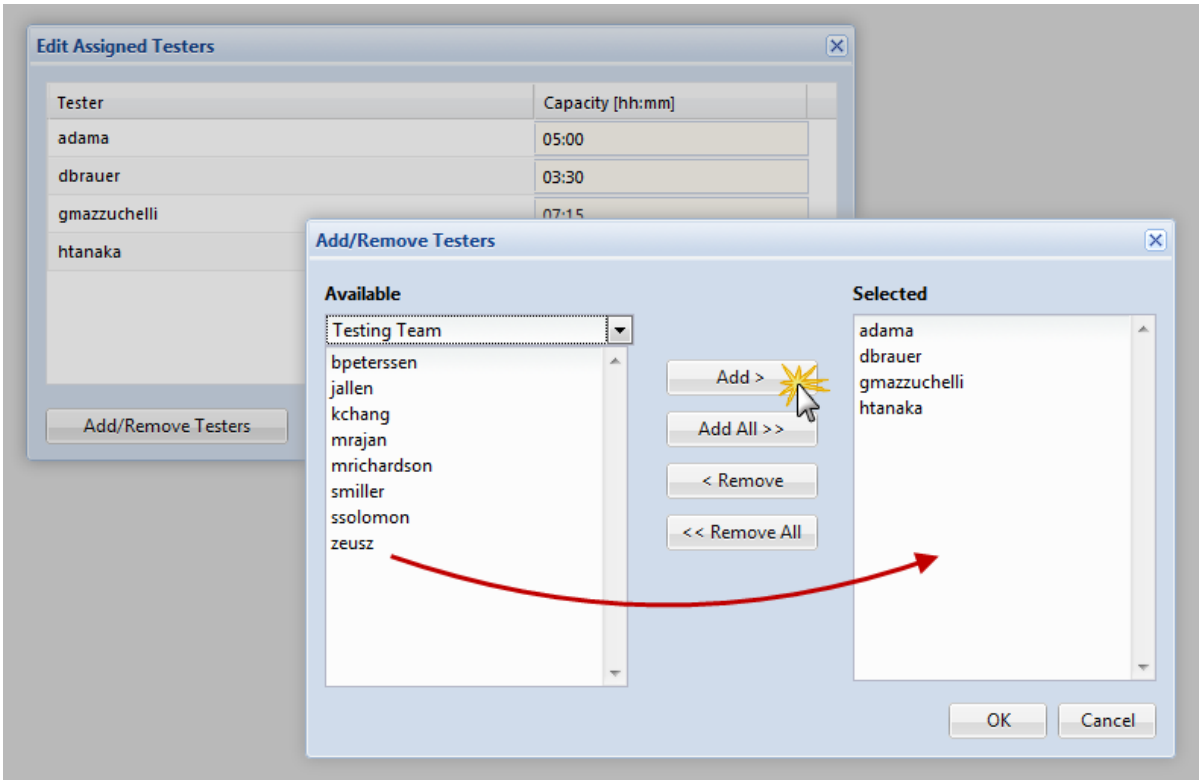


Adding Manual Testers (Walkthrough)

The next step is to add manual testers to our testing cycle. We are also going to define the capacity each manual tester has for this testing cycle.

1. In the header of our testing cycle `Week 21 - Team A`, click > **Edit Assigned Testers**. The **Edit Assigned Testers** dialog box appears.
2. Click **Add/Remove Testers**. The **Add/Remove Testers** dialog box appears.
3. Select a user group from the **Available** list.
4. Select one or more users from the user group in the **Available** list.
5. Click **Add** or **Add All** to add users to the **Selected** list.
6. Click **OK**. The users are now added to the list in the **Edit Assigned Testers** dialog box.
7. For each user that was added, provide a capacity for the testing cycle in the **Capacity** field using the `hh:mm` format.
8. Click **Close**.

Now our testing cycle shows the available resources, both for the whole testing cycle as well as for each manual tester. A burn-down chart displays the assigned tests that are not yet completed. After we have created our testing cycle and added manual testers, we can determine which tests need to be done in the next step.

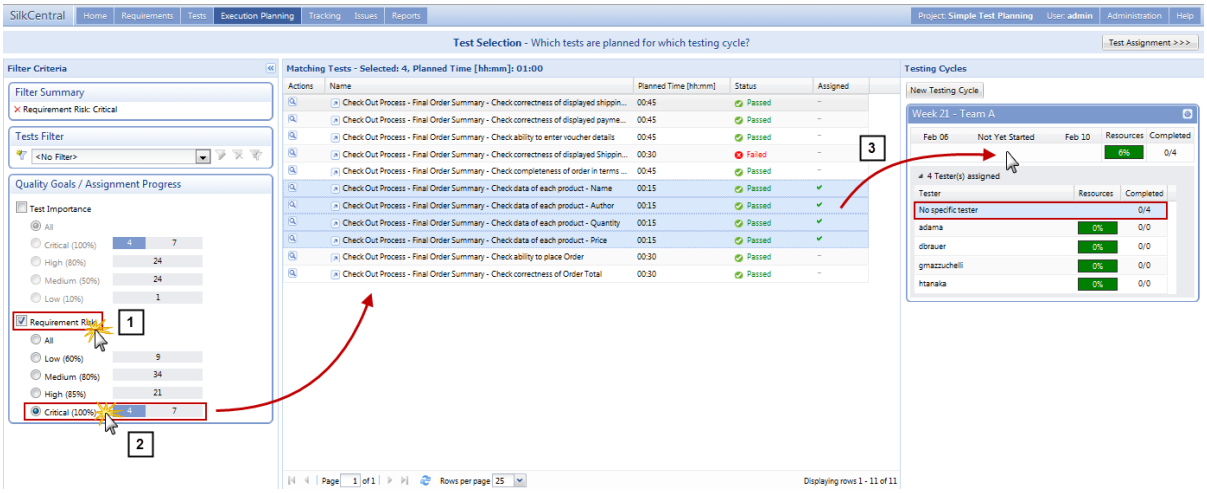


Selecting Tests and Scheduling (Walkthrough)

This topic describes how to assign tests to our testing cycle. The goal is to select those tests that best cover the defined quality goals. One of the quality goals is based on the requirement risk. The typical practice is to test things with high risk first.

1. In the **Filter Criteria** area on the left side, under **Quality Goals**, check the **Requirement Risk** check box. The list of **Matching Tests** is narrowed down to the tests that are critical.
2. In the **Matching Tests** area, right click on a column to sort, filter or group tests. That helps to select the tests that should be scheduled.
3. Select one or more tests, drag them to the **Testing Cycle** area on the right side and drop them to our testing cycle. When adding tests to a testing cycle, the indicator for already used resources is updated. Move the mouse over this indicator for detailed information about the overall capacity, the planned time, and the time left.
4. After scheduling all tests with a critical risk, schedule those with a high risk. After that, schedule those with a medium risk, and so on.

The tests assigned to our testing cycle have moved to the **No specific tester** row. In the next step, we will assign tests to manual testers.

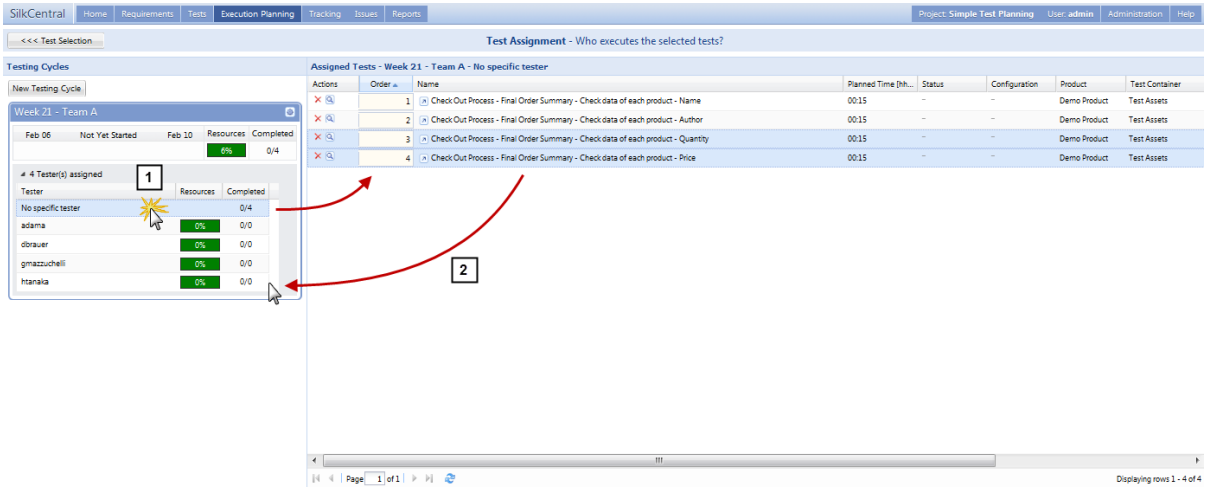


Assigning Tests to Manual Testers (Walkthrough)

Now we are going to assign the tests in our testing cycle to manual testers.

1. Click **Test Assignment >>>** on the top right. The **Test Assignment** screen appears.
2. In the **Testing Cycles** area on the left side, click on the **No specific tester** row in our testing cycle. All tests that are not assigned to a tester appear in the **Assigned Tests** area on the right side.
3. Select one or more tests, drag them to the **Testing Cycles** area on the left side and drop them to a manual tester. When assigning tests to a manual tester, the resource indicator is updated. Move the mouse over this indicator to see detailed information about the capacity, the planned time, and the time left.
4. When you are finished assigning tests to manual testers, click > **Start Testing Cycle** to start the tests.

When the testing cycle is started, each manual tester is informed through email about the tests that are assigned. You can add a test inbox, (**Manual Tests Assigned to Me**), to your personal dashboard. All the tests that are assigned to you are listed there.


SilkCentral
Test Assignment - Who executes the selected tests?
Testing Cycles
Assigned Tests - Week 21 - Team A - No specific tester
1
2

Configuration Testing

Configuration testing is the process of testing a system with each of the supported software and hardware configurations.

The **Execution** area supports configuration testing by allowing reuse of the assigned tests. You can create configuration suites with a set of assigned tests, and all execution plans that you add to the configuration

suite will also have the set of tests assigned. You can also create configuration suites from existing execution plans and copy and paste or cut and paste execution plans in the **Execution** tree into a configuration suite. Test Manager enables you to add or remove parameters, keywords, and manual testers to or from the configurations. When you create a configuration suite out of an existing execution plan, all the results of the execution plan are preserved in the configuration suite. If you copy-paste an execution plan into an existing configuration-suite, these results are not preserved.

Each execution plan in the configuration suite is displayed in an editable grid. You can view the execution plans or configurations that contain a specific test in the **Properties** tab of the test. You can also view the execution plans or configurations that are associated with a specific requirement in the **Assigned Tests** tab of the requirement.

To define configurations for automated tests, use the **Configurations Suite Configurations** page. To define configurations for manual tests use the **Configurations** page.

Creating a Configuration Suite

To create a configuration suite:

1. In the menu, click **Execution Planning** > **Details View** or **Execution Planning** > **Document View**.
2. Right-click the root node or a folder in the **Execution** tree and select **New Child Configuration Suite** or click in the toolbar. The **New Configuration Suite** dialog box displays.
3. Type a **Name** and **Description** for the new configuration suite.
4. Select a **Test Container** from the list box.
5. Select a **Version** and **Build** from the list boxes, or check the **Read from Build Information File** check box to read the version and build from a file.
6. *Optional:* In the **Source Control Label** text box you can specify that earlier versions of automation files, instead of the latest versions, are fetched from the source control system.

   Note: The **Source Control Label** property is only enabled if the associated test container uses a source control profile that supports versioning.
7. Click **OK**.

Creating a Configuration Suite from an Execution Plan

To create a configuration suite from an existing execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Right-click the execution plan, from which you want to create a configuration suite, in the **Execution** tree and select **Convert to Configuration Suite**. Test Manager creates a new configuration-suite node, with the same name as the selected execution plan, in the **Execution** tree and adds the execution plan as a configuration sub-node to the configuration suite.

When you create a configuration suite out of an existing execution plan, all the results of the execution plan are preserved in the configuration suite. If you copy-paste an execution plan into an existing configuration-suite, these results are not preserved.

Adding Parameters to a Configuration

To add a predefined parameter to a configuration:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution** tree, select the configuration node to which you want to add a parameter.
3. Click the **Configurations** tab.
4. Click **Add/Remove Parameters**. The **Add/Remove Parameters** dialog box opens. The **Assigned parameters** list box on the right side of the dialog box lists all of the parameters that are defined for the configuration.

5. Select parameters in the **Select or enter parameters** list box that reflect your execution environment requirements.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple parameters using standard browser multi-select functions.
6. To add any of the parameters listed in the **Select or enter parameters** list box, select the parameter and click **>**.
7. To create a new parameter, type the name of the parameter in the **Enter parameters** text box, and press **Enter** or click **>**.

   Tip: The **Enter Parameters** text box is auto-complete enabled. When you enter alphanumeric characters into this field, the field is dynamically updated with an existing parameter that matches the entered characters. The text box is disabled when multiple parameters are selected in the **Select parameters** or **Assigned parameters** list boxes.
8. Click **OK**. The **Add/Remove Parameters** dialog box closes and a new column is added to the grid for each new parameter.

Removing Parameters from a Configuration

To remove a predefined parameter from a configuration:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution** tree, select the configuration node from which you want to remove a parameter.
3. Click the **Configurations** tab.
4. Click **Add/Remove Parameters**. The **Add/Remove Parameters** dialog box opens. The list box on the right side of the dialog box lists all of the parameters that are defined for the configuration.
5. To remove a parameter from the list box, select the parameter and click **<**.
6. Click **OK**. The **Add/Remove Parameters** dialog box closes and the column of the removed parameter is removed from the grid.

Assigning Keywords to a Configuration

To assign keywords to a configuration:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the configuration suite that you want to edit.
3. Click the **Configurations** tab.
4. Click into the **Keywords** column of the configuration that you want to add a keyword to. The **Assign Keywords** dialog box opens.
5. Select keywords in the **Select keywords** list box that reflect your execution environment requirements.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple keywords using standard browser multi-select functions.

   Tip: The **Enter keywords** text box is auto-complete enabled. When you enter alphanumeric characters into this field, the field is dynamically updated with an existing keyword that matches the entered characters. The text box is disabled when multiple keywords are selected in the **Select keywords** or **Assigned keywords** list boxes.

   Tip: For automated execution plans, if you only have a few execution servers and do not require hardware provisioning, it might be enough to use only the default, reserved keywords that are created for each execution server. In such cases, it is not necessary that you select additional keywords.
6. Click **>** to move the keyword into the **Assigned keywords** list box.

   Note: For automated execution plans, the execution servers that match the assigned keywords are listed below in the dynamically-updated **Matching execution servers** list box. This list

updates each time you add or remove a keyword. Click on the name of an execution server in the list to access the execution servers in **Administration** > **Execution Servers** .

7. Click **OK** to save the keywords and close the **Assign Keywords** dialog box.

Removing Keywords from a Configuration

To remove keyword assignments from a configuration:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the configuration suite that you want to edit.
3. Click the **Configurations** tab.
4. Click into the **Keywords** column of the configuration that you want to remove a keyword from. The **Assign Keywords** dialog box opens.
5. Select the no longer needed keywords in the **Assigned keywords** list box.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple keywords using standard browser multi-select functions.
6. Click **<** to remove the keyword assignments.
7. Click **OK** to save the keywords and close the **Assign Keywords** dialog box.

Note: Keywords that are not in use anymore are automatically deleted from the system.

Adding Manual Testers to Configurations

For configurations that include manual tests, the **Configurations** page enables you to assign users as manual testers to a selected configuration. You can assign multiple manual testers to the configuration.

To assign a manual tester to a configuration:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the configuration suite that you want to edit.
3. Click the **Configurations** tab.
4. Click into the **Manual Testers** column of the configuration that you want to add a manual tester to. The **Manual Testers** dialog box opens.
5. In the **Available** list box, select the user group of which the tester is a member. The list box is populated with all members of the user group.
6. Select the name of the user you want to assign as a manual tester.
7. Click **Add** to add the user to the **Selected** list box; or click **Add All** to add all of the group's members and testers.
8. Click **OK** to close the **Manual Testers** dialog box.

Removing a Tester Assignment from a Configuration

To remove a tester assignment from a configuration:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the configuration node from which you want to remove a tester assignment.
3. Click the **Configurations** tab.
4. Click into the **Manual Testers** column of the configuration that you want to remove a manual tester from. The **Manual Testers** dialog box opens.
5. From the **Selected** list box, select the name of the assigned user that you want to remove.
6. Click **Remove** to remove the user from the **Selected** list; or click **Remove All** to remove all tester assignments.
7. Click **OK** to close the **Manual Testers** dialog box.

Configurations Suite Configurations Page

To access this page, click **Execution Planning** > **Details View** > **<Configuration Suite>** > **Configurations**.

The **Configurations** tab displays the configurations that are included in a configuration suite. The grid provides data for the automated generation of child execution plans.

Each configuration that you add to the grid results in a new execution plan, which is executed with the configuration suite.

To define a new configuration, click **New Configuration**. To add or remove a parameter, click **Add/Remove Parameters**.

For each listed configuration, the page displays the following columns:

Column	Description
Actions	• Click ✕ to delete the configuration. • Click to duplicate the configuration.
Name	The name of the configuration. Click on the name to change it.
Keywords	The list of the execution keywords that are assigned to the configuration. Click on the link to change the list.
Manual Testers	The list of the manual testers that are assigned to the configuration. Click on the link to change the list.
Parameters	The grid can have multiple parameter columns. Click **Add/Remove Parameters** to add or remove parameters from the grid. To add a value to a parameter, click the field and type in the value.

Configurations Page

To access this page, click **Execution Planning** > **Configurations**.

The **Configurations** page shows a list of the defined configurations in a grid. You can use these configurations for manual tests only. To define configurations for automated tests, see *Configurations Suite Configurations Page*.

To define a new configuration, click **New Configuration**. To add or remove a parameter, click **Add/Remove Parameters**.

For each listed configuration, the page displays the following columns:

Column	Description
Actions	• Click ✕ to delete the configuration. • Click to duplicate the configuration.
Name	The name of the configuration. Click on the name to change it.
Keywords	The list of the execution keywords that are assigned to the configuration. Click on the link to change the list.
Parameters	The grid can have multiple parameter columns. Click **Add/Remove Parameters** to add or remove parameters

Column	Description
	from the grid. To add a value to a parameter, click the field and type in the value.

Execution Planning Document View

Execution Planning > **Document View**

The **Execution Planning** area offers two views of execution properties: **Document View** and **Details View**. **Document View** provides a high-level, read-only view of the defined properties and characteristics of the latest run of all execution plans, folders, and configuration suites in the selected project. Use the **Document View** to plan the execution of your execution plans. You can use filters to obtain an execution-duration estimate of the filtered execution-plan subset.

The **Document View** displays the following columns for a selected element in the **Execution Plans** tree:

Column	Description
Status	The cumulated test statuses of the latest run of the execution plan, or the cumulated test statuses of all contained execution plans, for a folder or configuration suite. If you apply a filter to the **Execution Plans** tree, the test status totals are based on the filtered execution plans only.
Assigned Tests	Total number of assigned tests. If you apply a filter to the **Execution Plans** tree, the amount of the assigned tests is based on the filtered execution plans only.
Build	Build that is used when the execution plan is executed.
Version	Version that is used when the execution plan is executed.
Product	Product that the associated test container is based on.
Priority	For automated execution plans, when more than one execution plan is queued but only one execution server is available, the **Priority** determines which execution plan is executed first.
Last Execution	Last time the execution plan was started.
Planned Time	Total planned time for all manual tests. If you apply a filter to the **Execution Plans** tree, the planned time is based on the filtered execution plans only.
Duration	Duration of the latest run of an execution plan, or in case of a folder or configuration suite, the cumulated durations of the latest runs of all contained execution plans. If you apply a filter to the **Execution Plans** tree, the duration is based on the filtered execution plans only.
Next Execution	Next scheduled execution.
Test Container	Test container containing the tests that can be assigned to this execution plan.

Note: Execution plan totals accumulate to the parent level, for all currently displayed execution plans. If you apply a filter to the **Execution Plans** tree, the execution plan totals are based on the filtered execution plans only. For example, the execution plan totals include execution runs from all child execution plans that are selected with the filter, and the project totals include execution runs from all execution plans that are selected with the filter.

Working with the Execution Plans Tree

Execution Planning > **Details View**

Folders, execution plans, and configuration suites are displayed, organized, and maintained through a hierarchical tree structure, the **Execution Plans** tree. The **Execution Plans** tree enables you to organize folders, execution plans, and configuration suites in any number of hierarchy levels.

Note: When the **Execution Plans** tree includes more elements than can be displayed at once without impacting response time, the elements are displayed in increments. Page number links at the lower part of the tree allow you to browse through the elements included on the tree one page at a time. To display all elements as a single list, click the **[All]** link.

Expanding or Collapsing the Execution Plans Tree

You can consolidate levels of the **Execution Plans** tree or display all levels of the hierarchy based on your viewing needs.

To collapse or expand levels of the **Execution Plans** tree:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select a folder in the **Execution Plans** tree and continue with one of the following options:
   - Click ▷ to the left of the name of the folder to expand the folder.
   - Click ◢ to the left of the name of the folder to collapse the folder.
   - Right-click on the folder and select **Expand** or **Collapse**.

Editing Elements of the Execution Plans Tree

To edit an existing execution plan element:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan, folder, or configuration suite you want to edit.
3. Click on the toolbar or right-click the element and select **Edit**. A dialog box displays.
4. Edit the element by modifying the criteria in the dialog box.
   For execution plans, if there are no runs and no tests assigned to the execution plan, you can choose an alternative test container for the execution plan from the **Test Container** list box.
5. Click **OK** to save the edited element.

Copying Elements of the Execution Plans Tree

To copy and paste an execution plan, a folder, or a configuration suite:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the element that you want to copy.
3. Click on the toolbar or right-click the element and select **Copy** .
4. Select the target folder where the element is to be pasted.
5. Click on the toolbar or right-click the target node and select **Paste**. The **Execution Plans** tree is updated with a copy of the pasted element. All assigned tests, filters, and scheduling parameters are copied along with the element.

Deleting Elements of the Execution Plans Tree

To delete an execution plan, a folder, or a configuration suite:

1. In the menu, click **Execution Planning** > **Details View**.

2. In the **Execution Plans** tree, select the element that you want to delete.
3. Click ✕ on the toolbar or right-click the element node and select **Delete**.
4. Click **Yes** on the deletion confirmation dialog box to remove the element from the **Execution Plans** tree.

When deleting an element of the **Execution Plans** tree, the run results of the assigned tests are also deleted. The test run results may still appear in reports, because they are stored in the database, which is not immediately updated after the deletion of the element.

Filtering the Execution Plans Tree

Follow the steps below to filter the **Execution Plans** tree so that only the selected node and child nodes are visible. This is useful if you want to print a sub-set of execution plans to a PDF.

1. Create a custom filter that defines the execution plans that you want to view.
2. Select a folder or configuration suite in the tree view, right click and select **Filter Subtree**.

Execution Plan Properties Page

Execution Planning > **Details View** > **<Execution Tree Element>** > **Properties**

The **Properties** page lists basic properties that are relevant to the selected project, folder, execution plan, or configuration suite.

Property	Description
Execution Plan Name/Configuration Suite Name/ Folder Name	Name of the element in the **Execution Plans** tree.
Execution Plan ID/Configuration Suite ID/Folder ID	Database identifier of the element.
Description	Description of the element. **Note:** Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.
Test Container	The test container with witch the execution plan or configuration suite is associated. Click to access the test container in the **Tests** area.
Version	Version that is used when the execution plan is executed.
Build	Build that is used when the execution plan is executed.
Priority	For automated execution plans, when more than one execution plan is queued but only one execution server is available, the **Priority** determines which execution plan is executed first.
SilkTest Classic AUT Host NameSource Control Label	Hostname of the application under test. For SilkTest Classic execution plans only.
Tests	The tests associated with this execution plan or configuration suite.
Source Control Label	Not existing for configuration suites. In the **Source Control Label** text box you can optionally specify that the execution plan is of an earlier version than the latest version. The label must refer to a version in the source control system that the test container is associated with. If

Property	Description
	this text box is left blank, the latest version of the execution plan will be fetched. The **Source Control Label** property is only enabled if the associated test container uses a source control profile that supports versioning. Make sure to have enough free disk space on the execution server or servers when working with multiple versions of source files. Each version will be saved in its own folder on every execution server.
Status of last run	The cumulated test statuses of the latest run of the execution plan, or the cumulated test statuses of all contained execution plans, for a folder or configuration suite.
Last Execution	Last time an execution of the element was started.
Duration	Duration of the latest run of an execution plan, or in case of a folder or configuration suite, the cumulated durations of the latest runs of all contained execution plans. In the simplest case, automated tests on a single execution server or only manual tests, the duration is the time displayed for the latest run on the **Runs** page. If the last execution involved both automated and manual tests, only the automated or manual tests will be considered, depending on which test was executed later. If the execution plan contains multiple tests, the duration is measured from the time when the first test begins executing and ends when the last test completes execution. This includes the overhead time, which is needed for stopping/starting tests between executions.
Next Execution	Next time this element will be executed.
Created On	Date and time the element was created.
Created By	Name of the user who created the element.
Changed On	Date and time the element was last changed.
Changed By	Name of the user who last changed element.

Testing Cycle Properties Page

Execution Planning > **Details View** > **<Testing Cycle>** > **Properties**

The **Testing Cycles Properties** page is available when you select any testing cycle from the **Execution Plans** tree and then click **Properties**.

Click **Edit Properties** to display the **Edit Testing Cycle** dialog box where you can edit the **Name**, **Description**, and **Planned Time** of the testing cycle.

The following table displays the available testing cycle properties.

Property	Description
Testing Cycle Name	Name of the testing cycle. Edit this value in the **Edit Testing Cycle** dialog box.
Testing Cycle ID	Unique ID of the testing cycle.

Property	Description
Description	Description of the testing cycle. Edit this value in the **Edit Testing Cycle** dialog box.
Status of Last Run	Status is an aggregate of the last run of all test statuses within the execution plans of the testing cycle: **Passed**, **Failed**, **Not Executed**, or **N/A**.
Start Time of Last Run	Start time of the last run.
Testing Cycle Start Date	The scheduled start date for the testing cycle.
Testing Cycle End Date	The scheduled end date for the testing cycle.
Capacity [hh:mm]	Amount of time that you expect the testing cycle to take. Edit this value in the **Edit Testing Cycle** dialog box. If you assign manual testers to the testing cycle, the value for **Capacity** is the sum of the **Capacity** for each manual tester.
Planned Time [hh:mm]	Cumulative planned time of all manual tests within all assigned execution plans of the testing cycle.
Time Left [hh:mm]	This is the remaining time available after subtracting **Planned Time** from **Capacity**.
Created On	Date that the testing cycle was created.
Created By	User that created the testing cycle.
Changed On	Date that the testing cycle was last modified.
Changed By	User that last modified the testing cycle.

Assigned Manual Testers

This section lists all of the manual testers included in the cycle.

Property	Description
Tester	User name of the tester.
Capacity	Enter the amount of time that this tester is available for the testing cycle in hh:mm format. The sum of all capacity values for each tester will be the testing cycle **Capacity** value.
Add/Remove Testers	Click to display the **Manual Testers** dialog box which you use to add testers to the testing cycle.

Managing Execution Plans, Folders, and Configuration Suites

An execution plan is a collection of assigned tests that are stored in a single test container. An execution plan can be run at configurable schedules and deployed on specified execution servers.

The process of adding and editing execution plans is the same for both automated execution plans and manual execution plans.

Creating Execution Plans

To create an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select an existing folder or testing cycle in the **Execution Plans** tree, or select the project node.
3. Click on the toolbar or right-click the selected node and choose **New Child Execution Plan**. The **New Execution Plan** dialog box displays.
4. Type a name and description for the execution plan.

Note: Test Manager supports HTML formatting and cutting and pasting of HTML content for **Description** text boxes.

5. Select a test container from the **Test Container** list box.

   You can only associate one test container to an execution plan, and you can only execute tests from this single test container.

   The latest version and build that are defined within the product that the container belongs to are populated automatically in the **Version** and **Build** list boxes.
6. Select a product **Version** and **Build** from the list boxes.

   These are used when a new run of this execution plan is started. Alternatively, you can check the **Read from Build Information file** check box, in which case the version and build are read, at the beginning of each run, from the build information file on the application server. If a build information file is available on the execution server, this file is used by default for the test run, overriding the settings on the **New Execution Plan** dialog box.
7. Select a **Priority** for the execution plan from the list box.

   The priority parameter specifies the lowest priority that is considered in the data.
8. *Optional:* In the **Source Control Label** text box you can specify that earlier versions of automation files, instead of the latest versions, are fetched from the source control system.

   Note: The **Source Control Label** property is only enabled if the associated test container uses a source control profile that supports versioning.
9. Click **OK** to update the **Execution** tree with the newly created execution plan.

   Note: When you create data-driven tests, and a test has each data row as a single test, the execution plan includes a separate test for each data row. To create an execution plan with only a selection of data-driven tests, you need to assign tests with the filter option.

Creating an Execution Plan in Grid View

To create an execution plan in **Grid View**:

1. In the menu, click **Tests** > **Grid View** .
2. Select the tests you want to assign to your execution plan, by using the multi-select feature of the **Grid View**.
3. Right-click the tests and choose **Create Execution Plan**. The **New Execution Plan** dialog box displays.
4. Enter the specifications of your new execution plan.

Note: All selected tests must be in the same container. If not, the execution plan is not created and an error message displays. The test container is preselected in the **New Execution Plan** dialog box and can not be altered.

Test Status Calculation

Each execution plan has one of the following status conditions:

Status	Condition
`Passed`	All considered test execution plans have the status `Passed`.
`Failed`	At least one considered test execution plan has the status `Failed`, but none of the execution plans has the status `Not Executed`.
`Not Executed`	At least one considered test has the status `Not Executed`.

A test gets its status from the result of the latest execution plan run. If you manually change the status of the latest execution plan run, the test status changes also.

Note: If the latest execution plan run is deleted, the status of the test resets to the status of the latest existing execution plan run. If the deleted execution plan run was the only existing execution plan run, the status of the test is set to `N/A`, as if the test was newly created.

Execution Plan Parameters

To apply a specific configuration to an execution plan, you can override the parameters that you have assigned to a test in the **Parameters** page of the execution plan. For additional information on test parameters, see *Test Parameters*.

Creating an Execution Plan Parameter

To add a new parameter to an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution** tree, select the execution plan to which you want to add a new parameter.
3. Click the **Parameters** tab.
4. Click **New Execution Plan Parameter**. The **New Execution Plan Parameter** dialog box displays.
5. Type the **Name** and **Value** for the new parameter into the corresponding text boxes.
6. Click **Save** to add the parameter to the execution plan.

Note: You cannot create execution plan parameters with the same name as an existing test parameter, which is included in the execution plan.

Overriding a Test Parameter

If there exist more than one parameters with in the same name and different types in the tests, you can override only one of them. Only the values of parameters with the same type and name are replaced with the overriding value during execution.

To override a test parameter in an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution** tree, select the execution plan in which you want to override a test parameter.
3. Click the **Parameters** tab.
4. In the **Actions** column of the parameter that you want to override, click . The **Set Execution Plan Parameter Value** dialog box displays.
5. Type the new **Value** into the text box.
6. Click **OK**.

Note: Click in the **Actions** column of the parameter to undo the override.

Removing an Execution Plan Parameter

Note: You cannot remove test parameters in an execution plan. You can only remove execution plan parameters.

To remove an execution plan parameter:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution** tree, select the execution plan from which you want to add a new parameter.
3. Click the **Parameters** tab.
4. Click .

Execution Parameters Page

Execution Planning > **Details View** > **<Execution Plan>** > **Parameters**

The **Parameters** page displays the parameters of all tests that are assigned to the execution plan. You can override the parameters of the tests for the execution plan. For configuration testing, you can create multiple execution plans that include the same tests and you can then define different configurations in the **Parameters** page. Parameters from different tests that have the same name and type are displayed only once in the page. If multiple parameters from different tests have the same name, you can only override one of these parameters.

Click **New Execution Plan Parameter** to define a new execution plan parameter that is used for the execution of all tests that are included in the execution plan. New execution plan parameters are always of the type string and you cannot define a different type.

The page displays the following columns for each parameter:

Column	Description
Actions	Actions you can perform on the parameter. Click to edit the parameter value, or click to undo your changes.
Name	Name of the parameter.
Type	A parameter can have one of the following types: • String • Number • Float • Boolean • Password • Character
Value	The parameter value.

Setup and Cleanup Tests

The pre-test setup and post-test cleanup functionality of Test Manager enables you to define a setup test and a cleanup test for each execution plan. Setup tests are typically built upon scripts or manual procedures that prepare testing environments in anticipation of tests. Cleanup tests typically include scripts or manual procedures that restore test environments to their original state following tests. When setup tests are aborted, regular tests do not execute, however cleanup tests do execute to restore the testing environment to its original state.

You must create your setup and cleanup tests before you can assign them to execution plans. Any test can serve as a setup or cleanup test, except parent tests of multiple data-driven test instances. They require no special configuration and can be either automated or manual. The only requirement is that they perform the required setup and cleanup processes within your test environment. In the case of automated tests these are scripts that perform required setup and cleanup tasks. In the case of manual tests, these are manual setup and cleanup tasks.

Note: The challenge in executing setup and cleanup tests is preventing their results from being aggregated with the results of the regular tests that they support. Test Manager addresses this concern by running setup and cleanup tests, both automated and manual, in independent execution plans, thereby isolating actual test results from incidental performance fluctuations that may be caused by setup and cleanup tests.

Combining Automated and Manual Tests

Test Manager supports execution plans that include combinations of automated tests and manual tests. For such execution plans, Test Manager withholds execution of regular tests, both automated and manual,

until setup tests are complete. Test Manager also ensures that all regular tests are complete before cleanup tests are run.

When manual tests are combined with automated tests, automated tests, on all execution servers, do not begin until the setup processes are complete. In the case of manual setup tests, regular automated tests begin only after manual setup routines are complete.

Configuring Setup and Cleanup Executions

To define a test as a setup or cleanup test:

1. In the menu, click **Execution Planning** > **Details View**.
2. Click the execution plan for which you want to configure a setup or cleanup test.
3. Click the **Setup/Cleanup** tab.
   - To define a setup test, proceed with the following step.
   - To define a cleanup test, proceed with step 5.
4. *Optional:* To define a setup test, click **Edit** in the **Setup Test** section. The **Edit Setup Test** dialog box displays.
5. To define a cleanup test, click **Edit** in the **Cleanup Test** section. The **Edit Cleanup Test** dialog box displays.
6. In the **Tests** tree, select the test that you want to configure.
7. Click **OK**.

The configured test displays in the corresponding section of the **Setup/Cleanup** page.

Execution Setup/Cleanup Page

Execution Planning > **Details View** > **<Execution Plan>** > **Setup/Cleanup**

Note: This page is not displayed for execution plans that are assigned to a testing cycle.

The **Setup/Cleanup** page lists the setup and cleanup tests that are defined for this execution plan.

Note: When failed tests are rerun, the corresponding setup/cleanup routines are also rerun. Setup/cleanup tests are not run with **Try Run** test runs because such executions do not rely on execution plans.

Setup Test

In the **Setup Test** section, the page displays the following items:

Item	Description
Actions	**Delete** and **View Description**.
Test	Type and name of the configured setup test.
Edit	Opens the **Edit Setup Test** dialog box where you can select a setup test. You cannot assign a test to the same execution plan as both a setup test and a regular or cleanup test. Assigned tests can come from any test container within your project. It is therefore possible to assign tests that have associated products and source

Item	Description
	control profiles that vary from their host execution plans.

Cleanup Test

In the **Cleanup Test** section, the page displays the following items:

Item	Description
Actions	**Delete** and **View Description**.
Test	Type and name of the configured cleanup test.
Edit	Opens the **Edit Cleanup Test** dialog box where you can select a cleanup test. You cannot assign a test to the same execution plan as both a setup test and a regular or cleanup test. Assigned tests can come from any test container within your project. It is therefore possible to assign tests that have associated products and source control profiles that vary from their host execution plans.

Configuring Deployment Environments

Adding a SilkTest Classic AUT Host

For execution plans that run SilkTest Classic tests, you may have a setup where the SilkTest Classic agent is on a different computer than the execution server. In this case, you can define the location of the SilkTest Classic agent, the SilkTest Classic AUT (Agent Under Test) Hostname.

To add a a SilkTest Classic AUT host to the selected execution plan or edit the host:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan to which you want to assign the SilkTest Classic AUT host.
3. Click the **Deployment** tab.
4. In the **SilkTest Classic AUT Hostname** section, click **Edit**. The **Edit SilkTest Classic AUT Hostname** dialog box displays.
5. In the **Hostname** text box, type the name of the computer where the SilkTest Classic agent runs.
   Proper configuration of option files is required. For details on the command-line option `-m`, refer to the SilkTest Classic documentation.
6. Click **OK** to add the SilkTest Classic AUT host to the selected execution plan.

Adding Manual Testers

For execution plans that include manual tests, the **Deployment** page enables you to assign users as manual testers to a selected execution plan. You can assign multiple manual testers to the execution plan.

To assign a manual tester to an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Deployment** tab.

4. Click **Edit** in the **Manual Testers** section. The **Manual Testers** dialog box displays.
5. In the **Available** list box, select the user group of which the tester is a member. The list box is populated with all members of the user group.
6. Select the name of the user you want to assign as a manual tester.
7. Click **Add** to add the user to the **Selected** list box; or click **Add All** to add all of the group's members and testers.
8. Click **OK** to close the **Manual Testers** dialog box.

Removing a Tester Assignment from an Execution Plan

To remove a tester assignment from the selected execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan for which you are removing a tester assignment.
3. Click the **Deployment** tab.
4. Click **Edit** in the **Manual Testers** section. The **Manual Testers** dialog box displays.
5. From the **Selected** list box, select the name of the assigned user that you want to remove.
6. Click **Remove** to remove the user from the **Selected** list; or click **Remove All** to remove all tester assignments.
7. Click **OK** to close the **Manual Testers** dialog box.

Dynamic Hardware Provisioning with Keywords

The hardware-provisioning technology of Test Manager helps you manage test environments that include numerous execution servers. Rather than having to configure a one-to-one static execution-server assignment for each automated execution plan, keywords enable Test Manager to automatically select the most appropriate execution server for each execution plan. This is done through dynamic comparison of each execution plan's keyword list with the keyword lists of all active execution servers.

Keywords typically describe your execution environment requirements, for example platform, operating system, and pre-installed applications. There are different uses for keywords, depending on whether the execution plan is executed automatically or manually.

Automated execution plans	When an automated execution plan is executed, Test Manager compares the execution plan's keywords with the keywords of all available execution servers. The execution is then run on the first-identified execution server that has a matching keyword list.
Manual execution plans	For manual execution plans, the manual tester can reflect the test environment by using keywords.

If you require an automated execution plan to be run on multiple execution servers, create a copy of the execution plan and assign additional keywords to the execution plan that match other execution servers.

Reserved Default Keywords

If you do not require hardware provisioning to execute automated execution plans, you can use the reserved keywords that are created automatically for each execution server. In such cases, it is not necessary that you manually assign keywords to your execution servers. Instead, you can configure a one-to-one static execution-server assignment for each execution plan.

A reserved keyword is assigned automatically to each newly created execution server. Reserved keywords are structured in the following form: `#<execution server name>@<location name>`.

Reserved keywords are only available when assigning keywords to execution plans. They are neither available or applicable when assigning keywords to execution servers.

In addition to the reserved keywords that are set up automatically for each defined execution server, reserved keywords are also set up for each execution server type:

#PHYSICAL	Limits execution-server provisioning to physical execution servers.
#VIRTUAL	Limits execution-server provisioning to virtual execution servers.

Keywords and Virtual Execution Servers

Keywords are assigned to virtual execution servers in the same way that they are assigned to physical execution servers. When you configure at least one virtual execution server, the *#VIRTUAL* keyword is dynamically created and made available for assignment to all execution plans. If you prefer that an execution occur on a virtual machine, select the *#VIRTUAL* keyword for the execution plan. When an execution plan has neither the *#VIRTUAL* and *#PHYSICAL* keywords, the execution may occur on either a virtual or a physical execution server, assuming the settings of the execution environments are the same. When a test's keywords match multiple virtual execution servers, the first matching virtual execution server that is identified is selected.

Folder Execution

The execution plans can be combined into execution folders, where a folder can include execution subfolders and execution plans. The options for an execution plan execution are also available for an execution folder execution.

When executing a folder, the contained subfolders and execution plans are treated as follows:

Keywords of Executed Folder	Keywords of Contained Execution Plan/Subfolder	Execution of Contained Execution Plan/Subfolder
Has no keywords	Has no keywords	An execution plan without keywords obtain status NOT EXECUTED after execution
Has no keywords	Has keywords	Execution servers are assigned based on the execution plan/ subfolder keywords
Has keywords	Has no keywords	Execution servers are assigned based on the folder keywords
Has keywords	Has keywords	Execution servers are assigned based on the folder keywords

Note: When a folder is executed manually and there are no keywords assigned, or no execution server exists for the assigned keywords, the default execution server is used for execution. If the default execution server is not available, these execution plans are marked as `Not Executed`.

Creating New Keywords

To create new keywords:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Deployment** tab.
4. In the **Execution Environment** section of the page, click **Edit**. The **Assign Keywords** dialog box displays. All keywords that have been defined for your execution environment are listed here.

   Note: The default reserved keywords for each execution server, `#<execution name>@<location name>`, are included in the list.
5. Type an alphanumeric keyword into the **Keyword** text box that describes the required environment for the execution plan.

For example, the keyword might be the name of the platform, operating system, or pre-installed applications. The following characters can not be used in keywords: `#$?*\,;'"`.

Note: Keywords are case insensitive. For example, “Vista” and “vista” are handled as the same keyword.

6. Press `Enter`.

   The new keyword is now available for assignment.

Assigning Keywords to Execution Plans

To assign keywords to execution plans:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Deployment** tab.
4. In the **Execution Environment** section of the page, click **Edit**. The **Assign Keywords** dialog box displays. All keywords that have been defined for your execution environment are listed here.

   Note: The default reserved keywords for each execution server, `#<execution name>@<location name>`, are included in the list.

5. Select keywords in the **Select keywords** list box that reflect your execution environment requirements.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple keywords using standard browser multi-select functions.

   Tip: The **Enter keywords** text box is auto-complete enabled. When you enter alphanumeric characters into this field, the field is dynamically updated with an existing keyword that matches the entered characters. The text box is disabled when multiple keywords are selected in the **Select keywords** or **Assigned keywords** list boxes.

   Tip: If you only have a few execution servers and do not require hardware provisioning, it might be enough to use only the default, reserved keywords that are created for each execution server. In such cases, it is not necessary that you select additional keywords.

6. Click **>** to move the keyword into the **Assigned keywords** list box.

   Note: The execution servers that match the assigned keywords are listed below in the dynamically-updated **Matching execution servers** list box. This list updates each time you add or remove a keyword. Click on the name of an execution server in the list to access the execution servers in **Administration** > **Execution Servers** .

7. Click **OK** to save the keywords and close the **Assign Keywords** dialog box.

Removing Keywords from Execution Plans

To remove execution plan keyword assignments:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Deployment** tab.
4. In the **Execution Environment** section of the page, click **Edit**. The **Assign Keywords** dialog box displays. All keywords that have been defined for your execution environment are listed here.

   Note: The default reserved keywords for each execution server, `#<execution name>@<location name>`, are included in the list.

5. Select the no longer needed keywords in the **Assigned keywords** list box.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple keywords using standard browser multi-select functions.

6. Click **<** to remove the keyword assignments.

7. Click **OK** to save the keywords and close the **Assign Keywords** dialog box.

Note: Keywords that are not in use anymore are automatically deleted from the system.

Execution Deployment Page

Execution Planning > **Details View** > **<Execution Plan>** > **Deployment**

The **Deployment** page displays all of the hardware-provisioning keywords that are defined for this execution plan. These keywords are used to describe the execution environment requirements for the execution plan. An execution server only matches the selected automated execution plan if it has all keywords assigned the execution plan requires. The **Deployment** page also displays the users who are assigned to execute manual tests, as well as the SilkTest Classic AUT hosts that are assigned to execute SilkTest Classic tests.

Note: New execution servers are set-up in **Administration** > **Execution Servers**. For details on how to set-up an execution server, as well as configuring the integration of Test Manager with VMware Lab Manager, refer to the *SilkCentral Administration Module Help*.

Execution Environment

Note: This section is not displayed for execution plans that are assigned to a testing cycle.

The **Deployment** page displays the following items in the **Execution environment** section:

Item	Description	
Assigned keywords	Lists the keywords that are assigned to the execution plan.	
	Automated execution plans	Keywords are used to automatically identify an appropriate execution server for each test execution.
	Manual execution plans	Keywords are used by the manual tester to reflect the test environment.
Edit	Click to edit the execution plan's keywords.	
Matching execution servers	Lists the active execution servers that have keyword lists that match the keywords list of this execution plan. All keywords in the keywords list of the execution plan must be included in the keyword list of the execution server. Click on the name of an execution server in the list to access the execution server list in **Administration** > **Execution Servers** .	

Capturing Options

Note: This section is not displayed for execution plans that are assigned to a testing cycle.

Note: Only VMware Lab Manager configurations are captured. LiveLink URLs are attached to execution plan results as links on the **Messages** page and as separate html files that contain the LiveLinks.

The following VMware LiveLink capturing options are available for VMware Lab Manager configurations:

Option	Description
Never	Don't capture configurations.
Immediately on error	Once a failed test is completed, no further tests are executed and the configuration is captured.
After completing all tests	Upon failure conditions, continue test execution and capture the configuration after executing all tests of the execution plan.
Always	Capture configuration with each run of the test execution.

Manual Testers

Lists all manual testers who are assigned to this execution plan or folder. Click **Edit** to edit the list of manual testers.

SilkTest Classic AUT Hostname

Lists all SilkTest Classic AUT hosts that are defined for this execution plan. Click **Edit** to edit the list of SilkTest Classic AUT hosts.

Code Analysis Settings

Details code-analysis settings that are defined for this execution plan. Click **Active/Inactive** to enable or disable code analysis for this execution plan.

For virtual execution on VMware Lab Manager configurations, the internal IPs of the affected machines within the configuration must be configured here.

Video Recording

Note: This section is not displayed for execution plans that are assigned to a testing cycle.

Defines whether a video is recorded during the execution of the execution plan. This setting overrides the video recording settings of the individual tests that are assigned to the execution plan. You can select one of the following values:

Value	Description
---	Video recording is disabled, except for those assigned tests where the `recordVideo` is set.
Never	Video recording is always disabled, even for those assigned tests where the `recordVideo` is set.
Always	Records a video on every execution of the execution plan.
On Error	Records a video when an error occurs during the execution of the execution plan.

Screen Capturing

Note: This section is not displayed for execution plans that are assigned to a testing cycle.

Defines whether a screenshot is captured during the execution of the execution plan. This setting overrides the screen capturing settings of the individual tests that are assigned to the execution plan. You can select one of the following values:

Value	Description
---	Screen capturing is disabled, except for those assigned tests where the `captureScreen` parameter is set.

Value	Description
Never	Screen recording is always disabled, even for those assigned tests where the `captureScreen` parameter is set.
On Error	Captures a screenshot when an error occurs during the execution of the execution plan.

Configuring Execution Dependencies

An execution dependency allows you to configure the automatic execution of one execution plan based on the results of another execution plan. For example, *If execution plan 'A' fails, automatically execute execution plan 'B'*.

Adding Dependent Execution Plans

To add a dependent execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan that will act as the master execution plan.
3. Click the **Dependencies** tab.
4. Click **Add dependent Execution Plan**. The **Add dependent Execution Plan** dialog box displays.
5. From the **Condition** selection list, select the condition that is to trigger the dependent execution plan.
   - `Passed`
   - `Failed`
   - `Not Executed`
   - `Any`

   The `Any` status means that the dependent test execution will trigger no matter what the status of the previous test execution.
6. From the tree menu in the dialog box, select the execution plan that is to be dependent.
7. Specify where the dependent execution plan is to be deployed.

Option	Description
As specified in the dependent Execution Plan	Automated tests assigned to the dependent execution plan will be executed on the execution server specified for the dependent execution plan on the **Deployment** page. Manual tests assigned to the dependent execution plan will be assigned to the users specified for the dependent execution plan on the **Deployment** page.
Same as <selected execution plan's execution server>	Automated tests assigned to the dependent execution plan will be executed on the execution server specified for the master execution plan on the **Deployment** page. Manual tests assigned to the dependent execution plan will be assigned to the users specified for the master execution plan on the **Deployment** page.
Specific: Execution Server/ Manual Tester	Select a pre-configured execution server and/or a manual tester from the list boxes. Automated tests assigned to the dependent execution plan will be executed on the specified execution server. Manual tests assigned to the dependent execution plan will be assigned to the specified manual tester. If only a specific manual tester is defined and no server, only manual tests will be executed. If only a specific execution server is defined and no manual tester, only automated tests will be executed.

8. Click **OK** to create the dependency.

Note: Test Manager will not allow you to create cyclical execution dependencies. You can select conditions to fulfill for manual tests. For example if the selected condition is `Failed` and all

manual tests passed, but some automated tests failed, only automated tests assigned to the dependent execution plan are executed.

Editing a Dependency

Note: To edit an existing dependency, you must select the master execution plan, which is the definition for which a specific condition will trigger the execution of the dependent execution plan. You cannot edit dependency settings from the dependent execution plan.

To edit a previously configured dependency:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Dependencies** tab.
4. In the **Dependent Execution Plans** section, click in the **Actions** column of the dependent execution plan. The **Edit Dependency** dialog box opens.
5. From the **Condition** selection list, select the condition that is to trigger the dependent execution plan.
   - `Passed`
   - `Failed`
   - `Not Executed`
   - `Any`

   The `Any` status means that the dependent test execution will trigger no matter what the status of the previous test execution.
6. Specify where the dependent execution plan is to be deployed.

Option	Description
As specified in the dependent Execution Plan	Automated tests assigned to the dependent execution plan will be executed on the execution server specified for the dependent execution plan on the **Deployment** page. Manual tests assigned to the dependent execution plan will be assigned to the users specified for the dependent execution plan on the **Deployment** page.
Same as <selected execution plan's execution server>	Automated tests assigned to the dependent execution plan will be executed on the execution server specified for the master execution plan on the **Deployment** page. Manual tests assigned to the dependent execution plan will be assigned to the users specified for the master execution plan on the **Deployment** page.
Specific: Execution Server/ Manual Tester	Select a pre-configured execution server and/or a manual tester from the list boxes. Automated tests assigned to the dependent execution plan will be executed on the specified execution server. Manual tests assigned to the dependent execution plan will be assigned to the specified manual tester. If only a specific manual tester is defined and no server, only manual tests will be executed. If only a specific execution server is defined and no manual tester, only automated tests will be executed.

Deleting a Dependency

To delete a dependency:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the master execution plan from which you want to delete a dependency.
3. Click the **Dependencies** tab.
4. In the **Dependent Execution Plans** section, click in the **Actions** column of the dependent execution plan.
5. Click **Yes** on the **Delete Dependency** dialog box to delete the dependency.

Execution Dependencies Page

Execution Planning > **Details View** > **<Execution Plan>** > **Dependencies**

Note: This page is not displayed for execution plans that are assigned to a testing cycle.

The **Dependencies** page lists dependent execution plans and master execution plans of the selected execution plan.

The page is divided into two parts. For the selected execution plan, the page shows both the **Master Execution Plans**, which are the execution plans for which a specific condition triggers the selected execution plan, and the **Dependent Execution Plans**, which are the execution plans that will be triggered if the selected execution plan results in a specific condition.

Master Execution Plans

For each execution plan in the list, the page displays the following columns:

Column	Description
Name	Name of the master execution plan that the selected execution plan is dependent upon.
Condition	Condition of the master execution plan that must be met for the selected execution plan to be triggered.

Dependent Execution Plans

For each execution plan in the list, the page displays the following columns:

Column	Description
Name	Name of the dependent execution plan that the selected execution plan serves as the master of.
Condition	Condition of the selected execution plan that must be met for the dependent execution plan to be triggered.
Execution Server / User(s)	Execution server where the dependent execution plan is to be run or, in the case of a manual test execution, manual tester who is to perform the manual test.
Actions	Actions that can be performed on the selected dependency. **Edit Settings** and **Delete Dependency**.

Execution Notifications Page

Execution Planning > **Details View** > **<Execution Element>** > **Notifications**

The **Notifications** page includes check boxes which allow you to check whether you want to be notified based on the outcome of the execution.

Notification only works if an email server has been configured by your administrator. You also have to specify an email address for your account in **User: <User Name>** > **User Settings** . If notification has not been enabled, refer to the *SilkCentral Administration Module Help* or contact your administrator.

Check Box	Description
Execution finishing successfully	Check to receive a notification email each time an execution run finishes successfully.

Check Box	Description
Execution finishing with not passed tests	Check to receive a notification email each time an execution finishes with status `not executed` or `failed`.
Execution finishing with changed number of not passed tests	Check to receive a notification email each time the number of failed or not executed tests changes in comparison to the previous run, when an execution finishes.

Assigned Tests

This section describes how to assign tests to execution plans.

Assigning Tests from Grid View to Execution Plans

The tests that are assigned to the selected execution plans are listed on the **Assigned Tests** page.

To assign one or more tests from **Grid View** to one or more execution plans:

1. In the menu, click **Tests** > **Grid View** .
2. Select the tests you want to assign to execution plans.

   Use **CTRL + CLICK** or **SHIFT + CLICK** to select multiple tests using standard browser multi-select functions.
3. Right-click the selected tests and choose **Save Selection**.
4. In the menu, click **Execution Planning** > **Details View**.
5. Select the execution plan to which you want to assign the selected tests.
6. Click the **Assigned Tests** tab.
7. Click **Assign Saved Selection**.

Note: Only tests that reside in the test container of the execution plan are inserted. You can insert the selected tests to more than one execution plans. You can not insert them into requirements in a different project. The selection persists until you make a different selection or close Test Manager.

Manually Assigning Tests to Execution Plans

The tests that are assigned to the selected execution plans are listed on the **Assigned Tests** page.

To manually assign tests to an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan to which you want to assign the selected tests.
3. Click the **Assigned Tests** tab.
4. Click the **Manual assignment** option button. All tests of the test container which is associated with the selected execution are displayed in the **Tests** tree. If you have created a test filter, you can select it from the filter list above the **Tests** tree. To create a new test filter, navigate to the **Tests** area and click **New Filter** on the toolbar.
5. In the tree, click ⇦ left of the test that you want to assign to the currently selected execution plan.

   Clicking the assign arrow of a folder or the top-level container assigns all child tests of that parent to the selected execution plan.

Assigning Tests to Execution Plans through a Filter

The tests that are assigned to the selected execution plans are listed on the **Assigned Tests** page.

To use a filter to assign one or more tests to an execution plan:

1. Create a filter in the **Tests** area.
   See *Creating Filters* for details. If the filter already exists, skip this step.
2. In the menu, click **Execution Planning** > **Details View**.
3. Select the execution plan to which you want to assign tests.
4. Click the **Assigned Tests** tab.
5. Select **Assignment by filter** from the test assignment types.
6. Choose the filter from the list box.

Note: If you assign tests to an execution plan in **Tests** > **Grid View** , the test assignment type is automatically set to **Manual Assignment**, but the previously-filtered tests remain in the **Assigned Tests** page.

Locating Tests Assigned to Execution Plans

To locate manually assigned tests in the **Tests** tree:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan for which you want to locate the assigned tests.
3. Click the **Assigned Tests** tab.
4. *Optional:* Select **Manual assignment**, if it is not selected.
5. In the **Actions** column of a test, click to find out in which test folder or container the test is stored in.

The corresponding parent folder is expanded in the **Tests** tree and the assigned test is highlighted in blue.

Removing Test Assignments

To remove a manually assigned test:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan from which you want to remove the assignment.
3. Click the **Assigned Tests** tab.
4. In the **Actions** column of the assigned test, click .
   Repeat this step for all assignments that you want to delete.

   Tip: To remove all assigned tests, click **Remove All**.

Execution Assigned Tests Page

Execution Planning > **Details View** > **<Execution Element>** > **Assigned Tests**

Note: This page is not displayed for execution plans that are assigned to a testing cycle.

The **Assigned Tests** page lists all tests that are assigned to the selected execution plan or configuration suite. Use this page to assign additional tests to the execution plan or configuration suite, to remove tests from the execution plan or configuration suite, or to change the execution order of the assigned tests.

If you have created a test filter, you can select it from the filter list above the **Tests** tree. To create a new test filter, navigate to the **Tests** area and click **New Filter** on the toolbar.

Note: All changes in this page are immediately applied.

Note: When you access the page from a sub-element of a configuration suite, the page is read-only.

The page includes the following items:

Item	Description
Manual assignment	Click to manually assign tests to the execution plan or configuration suite.
Use test order	Check to set the execution order of the assigned tests to follow the execution order in the **Tests** area.
Assign Saved Selection	Click to assign a selection of tests from **Tests** > **Grid View**.
Assignment by filter	Click to automatically assign tests to the execution plan or configuration suite based on a pre-defined filter. The available filters are listed in the list box.
Assigned Tests	Amount of tests that are assigned to the execution plan or configuration suite.
Time Left in Testing Cycle [hh:mm]	This field is available when an execution plan is in a testing cycle. It indicates the hours remaining in a testing cycle after subtracting the combined planned time of all tests from the **Available Time** in the testing cycle. **Note: Planned Time** is for manual tests only.
Planned Time	The estimated time to execute all tests in the execution plan or configuration suite. For configuration suites, the **Planned Time** is the sum of the planned times of all the configurations in the suite. **Note: Planned Time** is for manual tests only.
Assigned Tests List	For each assigned test, the page displays the following columns: **Order** — The execution order of the test. Uncheck the **Use test order** check box to change the execution order. Click in the text box, type the new order of the test, and then press `Enter` to confirm the change. Each change in each of the text boxes must be confirmed by pressing `Enter`. If you change the orders of multiple tests without pressing `Enter` each time, just the last change before pressing `Enter` is taken. **Actions** — You can perform the following actions on the assigned tests when the **Manual assignment** option button is clicked: **Action** / **Description** Remove the selected test from the list. Locate the selected test in the test tree. **Test** — Name of the test. Click to access the test in the **Tests** area. **Status** — Status of the last run of the test in the context of the execution plan or the configuration suite. When the test is executed outside of the context of the execution plan or configuration suite, the displayed status remains unchanged. If the test has not yet been executed in the context of the execution plan or configuration suite, the status is `N/A`. For tests or test packages that are included in a configuration suite, the status is an aggregation of all statuses of all configurations within the suite: • If the test is assigned only to the configuration suite, the status is `Passed` only if it is passed in all configurations. If

Item	Description	
		the status is not `Passed` in all configurations, the status of the worst run is propagated, with the following top-down priority: **1.** N/A **2.** Not Executed **3.** Failed **4.** Passed The status of a requirement that is assigned to the test is `Failed` if the test fails in one or more of the configurations. • We do not recommend assigning the test to a configuration suite and one or more simple execution plans. However, in this case, the aggregate of the configuration statuses is one status, and the last statuses of the execution plans are other statuses. The status that was executed last determines the last status of the test. • If the test is assigned to a simple execution plan only, the status is updated in response to the status of the last test run.
	Last Execution	Date and time of the last run of the test in the context of the execution plan or configuration suite. When the test is executed outside of the context of the execution plan or configuration suite, the displayed time and date remain unchanged.
	Planned Time	The estimated time for the execution of the test. **Note:** Planned Time is for manual tests only.
Test	Shows all tests in the **Tests** tree that are available for assignment to the selected execution plan or configuration suite. Double-click a test or use ⇐ to assign the tests to the execution plan or configuration suite. For information about inserting multiple tests from the **Tests** area to the execution plan or configuration suite, see *Assigning Tests from Grid View to Execution Plans*.	

Schedules

In the **Schedule** page, once you have defined the tests that are to be included in an execution plan, a folder, or a configuration suite, you can define the schedule, by which the execution plan, folder, or configuration suite is to be executed.

Three scheduling options are available:

- **None**
- **Global**
- **Custom**

Note: You can define schedules for execution plans, folders, or configuration suites. If a schedule is defined for a folder or a configuration suite, all execution plans that are included in the selected folder or configuration suite will be executed at the specified schedule. All execution plans, folders, or configuration suites with no keywords assigned get the status `Not Executed` when executed in a

schedule. The scheduling options for parts of a project copy or baseline are set to none to prevent losing the last execution status. For more information on copies and baselines of projects, refer to the *SilkCentral Administration Module Help*.

Definite Runs

Definite runs enable you to define times at which tests will be executed regardless of configured schedules.

Creating a Custom Schedule

To create a custom schedule for a selected execution plan, folder, or configuration suite:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite for which you want to configure a custom schedule.

   Note: To save an edited version of a global schedule as a custom schedule, click **Edit** while the global schedule is selected in the list box. This enables you to edit the global schedule and save the result as a custom schedule.
3. Click the **Schedule** tab.
4. Click the **Custom** option button to enable the scheduling controls.
5. Click **Edit**.
6. Click next to the **From** field and use the calendar tool to specify the time and date when the execution schedule should begin.
7. Specify the **Interval** at which the tests should be executed.
8. In the **Run** section, specify when the schedule schould end.

   Select on of the following options:

   - Click **Forever** to define a schedule with no end.
   - Click **n Time(s)**.
   - Click next to the **until** field and use the calendar tool to specify the time and date when the execution schedule should end.
9. *Optional:* Click **Add Exclusion** to define times when scheduled elements should not be executed.
10. *Optional:* Click **Add Definite Run** to define times when unscheduled executions should be executed.
11. Click **Save** to save your custom schedule.

Specifying Global Schedules

Test Manager offers the possibility of defining global schedules, which can be reused in Test Manager for the scheduling of tests. Global schedules can speed up the process of scheduling tests, since the need to define individual schedules for each test is reduced to only those tests that require special scheduling.

To select a pre-defined schedule that is globally available throughout Test Manager:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite for which you want to configure a schedule.
3. Click the **Schedule** tab.
4. Click the **Global** option button.
5. Select the required pre-defined schedule from the list box.

   Details of the pre-defined schedule are displayed in a read-only calendar view. To save an edited version of a global schedule as a custom schedule, click **Edit**.

   Note: You can configure global schedules in **Administration** > **Schedules**.

Specifying No Schedule

To specify that no schedule should be defined for an execution plan, a folder, or a configuration suite:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite, for which you want to define that it is not to be executed based on schedules.
3. Click the **Schedule** tab.
4. Click the **None** option button.

Schedule Exclusions

A schedule exclusion is a regularly occurring time period during which executions should be suspended, for example during a weekly planned system downtime or during weekends. You can add as many exclusions to a schedule as you need. Exclusions enable you to define weekdays and time-of-day intervals during which tests are not to be executed, regardless of configured schedules. For example, you may not want tests to be executed on weekends.

Adding Exclusions

Note: You must have administrator rights to edit global schedules. To define a scheduling exclusion for a global schedule, navigate to **Administration** > **Schedules**.

To add an exclusion to a custom schedule:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select an execution plan, folder, or configuration suite, for which you want to add a scheduling exclusion.
3. Click the **Schedule** tab.
4. Click the **Custom** option button to enable the scheduling controls.
5. Click **Add Exclusion**.
6. On the **Configure Schedule Exclusion** page, select the weekdays on which tests should be suppressed.
7. Define the specific time intervals on those days during which execution should be suppressed.
8. Click **OK**. Your exclusion settings are now listed on the **Schedule** page.
9. Click **Save** to add the exclusion to the current schedule, or continue adding additional exclusions.

Editing Exclusions

To edit an exclusion:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite, for which you want to edit a previously configured exclusion.
3. Click the **Schedule** tab.
4. In the **Actions** column of the exclusion, click .
5. Edit the exclusion as required and click **OK**.
6. Click **Save**.

Deleting Exclusions

To delete an exclusion:

1. In the menu, click **Execution Planning** > **Details View**.

2. Select the execution plan, folder, or configuration suite, for which you want to delete a previously configured exclusion.
3. Click the **Schedule** tab.
4. In the **Actions** column of the exclusion, click .

Definite Runs

A definite run is a run of an execution plan, a folder, or a configuration suite, that you schedule to run at a specific time, independent of the schedule that is configured for the execution plan, folder, or configuration suite. You can add as many definite runs to a schedule as you need.

Adding Definite Runs

Note: You must have administrator rights to edit global schedules. To define a definite run for a global schedule, navigate to **Administration** > **Schedules**.

To add a definite run to a custom schedule:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite, for which you want to add a definite run.
3. Click the **Schedule** tab.
4. Click the **Custom** option button to enable the scheduling controls.
5. Click **Add Definite Run**.
6. On the **Configure Definite Run** page, click and select the date and time when the execution plan, folder, or configuration suite, should definitely be run.
7. Click **OK**. Your definite run settings are listed on the **Schedule** page.
8. Click **Save** to add the definite run to the current schedule, or continue adding definite runs.

Editing Definite Runs

To edit a definite run:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite, for which you want to edit a previously configured definite run.
3. Click the **Schedule** tab.
4. In the **Actions** column of the definite run, click .
5. Edit the definite run criteria as required and then click **Save**.

Deleting Definite Runs

To delete a definite run:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan, folder, or configuration suite, for which you want to delete a previously configured definite run.
3. Click the **Schedule** tab.
4. In the **Actions** column of the definite run, click .

Schedule Page

Execution Planning > **Details View** > **<Execution Element>** > **Schedule**

Note: This page is not displayed for execution plans that are assigned to a testing cycle.

The **Schedule** page is used to define schedules for execution plans, folders, and configuration suites.

The page provides the following schedule types:

Schedule Type	Description
None	Click this option button to define that the execution plan, folder, or configuration suite, is not executed based on a schedule. All schedules included in a copy or baseline of a project are initially set to none.
Global	Click this option button to select a pre-defined schedule from the list box for the execution plan, folder, or configuration suite. Selecting a global schedule includes the schedule exclusions and definite runs which are defined in the global schedule. For information on defining global schedules, refer to the *SilkCentral Administration Module Help*. Selecting a global schedule displays the schedule details below the **Custom** option button.
Custom	Click this option button to define a custom schedule for the execution plan, folder, or configuration suite. Click **Edit** to edit the custom schedule in the fields below.

For each selected schedule type, the page displays details. For a custom schedule, the details are editable.

Caution: If tests assigned to an execution plan, a folder, or a configuration suite, are not executed, the reason might be that the execution plan, folder, or configuration suite is still executed because of an earlier schedule, and the schedule interval is shorter than the duration of the execution. In this case either increase the schedule interval, improve the performance of the executed tests, or remove tests from the execution plan. To view the application server logfile, click the **Application Server Log** tab in **Administration** > **Log Files** .

Schedule Details

The page displays details for the selected schedule type, which are editable for a custom schedule.

Item	Description
From	Specifies when the execution schedule begins. Click next to the specified date to change the date and time.
Interval	Specifies the interval at which the tests are executed.
Adjust schedule to daylight savings	Check this check box to automatically have your schedule adjust to daylight savings time. **Note:** Daylight adjustment only works for intervals of two-hour multiples to avoid duplicate runs when setting time back one hour.
Run	Specifies when the execution ends: **Forever** Click this option button to specify that the execution should not stop executing.

Item	Description	
	Time(s)	Click this option button and select a number from the list box to define a specific number of executions.
	Until	Click this option button to pick a specific time at which test executions are to end. Click next to the specified date to change the date and time.
Exclusion	The exclusions that are defined for the schedule. To add an exclusion, click **Add Exclusion**.	
Definite Runs	The definite runs that are defined for the schedule. To add a definite run, click **Add Definite Run**. Click in the action column of a selected definite run to access the calendar tool and specify when the definite run is to take place.	

Executing Tests Assigned to Execution Plans

This section explains how to run tests that are assigned to execution plans with Test Manager.

You can run individual execution plans, folders, and configuration suites. In all cases you can define whether all assigned tests, or only tests matching certain status criteria should be executed. After an execution plan including automated tests is started, it is put in the central execution queue and an appropriate execution server is chosen to accomplish automated execution of the tests, based on the matching keywords and the server availability.

Starting Execution Plans

To run an execution plan independent of a schedule:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan, folder, or configuration suite that you want to execute.
3. Click **Run** on the toolbar. The **Run** dialog box displays.
4. Define which tests you want to execute in the **Run** dialog box.
5. If the execution plan does not contain pending manual tests, the **Go To Activities** dialog box displays. Click **Yes** to view the details of the execution plan runs in the **Activities** page, or click **No** if you want to remain on the current Web page.

   Note: Check the **Don't show this dialog again (during this login session)** check box if you do not want to be asked about switching to the **Activities** page again in the future. This setting will be discarded when you log out of Test Manager.

Run Dialog Box

Execution Planning > **Details View** > **<Execution Element>** > **Run**

Execution Planning > **Document View** > **<Execution Element>** > **Run**

The **Run** dialog box enables you to specify which tests you want to execute based on filter criteria and to specify which product build the test should be run against. To open the **Run** dialog box, select an execution plan or an execution folder and click **Run** on the toolbar.

<table>
<tr><th>Item</th><th>Description</th></tr>
<tr><td>All tests</td><td>Select this option to execute all tests.</td></tr>
<tr><td>Tests...</td><td>Select this option to only execute tests that meet one of the following options:<table><tr><th>Option</th><th>Description</th></tr><tr><td>With status Failed</td><td>Check to re-execute all tests in the selected execution plan, that have the status <code>Failed</code>.</td></tr><tr><td>Not executed with build ...</td><td>Check to re-execute all tests in the selected build that are not executed in the selected build. Tests that are executed in the selected or a later build are not re-executed.</td></tr></table></td></tr>
<tr><td>Tests that have had issues fixed since their last execution</td><td>Select this option to only execute those tests that have had issues advanced to the <code>Fixed</code> state since the test's last execution.</td></tr>
<tr><td>Set build for execution plan</td><td>Select a past build from the Set build for execution plan list box to have the test run against a specific past build. This field defaults to the current build. Note that this option is not available if the execution plan is configured to read the build number from a build information file. If an execution folder contains execution plans with different product versions assigned to each, the build cannot be selected for the execution of the execution folder.</td></tr>
<tr><td>Run Type</td><td>Choose Run as Specified to run all selected tests with their own test type, or choose Run automated tests manually to re-run all selected tests manually.</td></tr>
<tr><td>Go to Activities page</td><td>Check this check box to advance to the Activities page after you define tests for execution.</td></tr>
</table>

Manually Executing Automated Tests

Automated tests might sometimes deliver unexpected results because of environmental issues. When such a test fails in a system, and you want to verify if the failure is caused by a system component, or you know that the problem is caused by a currently dysfunctional system component, you can re-run the automated test manually.

To run an automated test manually, perform the following steps:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan, folder, or configuration suite that you want to execute.
3. Click **Run** on the toolbar. The **Run** dialog box displays.
4. Define which tests you want to execute.
5. Select **Run automated tests manually** in the **Run type** list box. All selected tests are then treated as manual tests, and the **Current Run** page opens.

Executing Manual Tests

Although manual tests are scheduled and managed alongside automated execution plans, their actual execution is quite different as they must be physically executed by a human tester.

To execute the manual tests that are assigned to you, click **Continue Manual Test** in the **Manual Tests Assigned to Me** panel in the **Dashboard**.

To distribute manual tests between the available manual testers and to schedule the execution of the manual tests, use the **Manual Execution Planning** page in the **Execution Planning** area.

Executing Manual Tests through the User Interface

Tip: If many users work on manual tests in parallel, we recommend to use the Manual Testing Client, an Eclipse-based client tool, to execute the manual tests.

To execute a manual test through the Test Manager user interface:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the manual execution plan that you want to execute.
3. Click **Run** on the toolbar. The **Run** dialog box displays.
4. Define which tests you want to execute and click **OK**.

   Tip: To go directly to the **Current Run** page, uncheck **Go to Activities page**.

   Note: When you choose the **Run** command on an execution plan, you must perform the manual test yourself. When you choose the **Run** command on a folder, all included manual tests within the folder must be executed by the testers who are assigned to the folder on the **Deployment** page, not the testers who are assigned to the individual execution plans.
5. If there are already pending manual tests in the selected execution plan, the **Manual Tests In Progress** dialog box displays.
   - Click **Start New** to create a new execution of the manual tests.
   - Click **Remove All Tests** to finish all pending manual tests and set their status to `Not Executed`.
   - Select a pending execution and click **Continue** to continue the selected execution.
6. On the **Current Run** page, proceed with the manual test execution.

Executing Manual Tests in the Current Run Page

The **Current Run** page allows easily finishing a manual test run. All required information is displayed in the two grid views, and the status of each test and test step can be changed with two clicks. You can see if someone is working on the test, in which case the tests status is `In Progress`, who is working on the test, and which test steps are already finished.

Tip: If many users work on manual tests in parallel, we recommend to use the Manual Testing Client, an Eclipse-based client tool, to execute the manual tests.

To execute a manual test using the current run page:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution** tree, select the execution plan with the assigned manual test that you want to execute.
3. Click **Run** on the toolbar. The **Run** dialog box displays.
4. Define which tests you want to execute and click **OK**.
5. The **Current Run** page opens.

   The page provides detailed information on every test step.
6. Click on the **Status** of a test step and change it to the appropriate status.

7. Repeat the previous step for all test steps.
8. *Optional:* Use **Ctrl + Click** and **Shift + Click** to select multiple test steps using standard browser multi-select functions. Right click on your selection and set the status of the selected test steps to the selected status.
9. *Optional:* Click **Finish Run** when you have completed testing of this execution plan. The **Finish Run** dialog box opens. Choose the appropriate build and the action to perform on the unfinished tests. The **Finish Run** dialog box opens. Choose the appropriate build and the action to perform on the unfinished tests.
10. The **Status** of the manual test is changed to the cumulated status of the test steps when the test run finishes. If one test step fails, the whole test is marked as failed.

Aborting Manual Test Executions

Tip: If many users work on manual tests in parallel, we recommend to use the Manual Testing Client, an Eclipse-based client tool, to execute the manual tests.

To abort a manual test execution:

1. In the menu, click **Tracking** > **Activities**.
2. In the **Current Executions** section, click ✕ in the **Actions** column of the manual test you want to remove.

Calculating the Status of a Manual Test Execution

When executing a manual test, the status of the test steps changes the overall status of the test.

The following statuses are available for a test step or test:

- `Not Executed`
- `Passed`
- `Failed`
- `Unsupported`
- `Unresolved`
- `In Progress` (only tests)

The calculation of the test status is based on the following rules, where the rules with lower numbers override the rules with higher numbers:

1. If the status of the test is `Not Executed`, and you change the status of a test step, the test status is set to `In Progress`.
2. As long as there is at least one step with status `Not Executed`, the test status remains `In Progress`.
3. If there is at least one step with status `Failed` or `Unresolved`, the test status is set to `Failed`.
4. If the status of every test step is `Passed` or `Unsupported`, the test status is set to `Passed`.

Printing Assigned Manual Tests

For each execution plan in the **Execution Plans** tree, you can print a document that includes information about all manual tests that are assigned to the execution plan. You can print this document to PDF or a local printer.

To print information about all manual tests that are assigned to an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, right click the execution plan and click **Print Assigned Manual Tests**. The **Manual Test Document** dialog box opens.
3. To print the assigned manual test information to a local printer, click **Print**. To save the information as a PDF document, click **Download as PDF**.

4. Click **Close** to close the **Manual Test Document** dialog box.

Video Capture in the Web UI

You can capture a video of a test execution in the Web UI which you can then upload to the selected test execution. You can then analyze the video if an error occurs during the test execution.

Java Runtime Environment (JRE) version 6 or higher is the recommended version to use video capturing. The JRE must be 32-bit. You can download JRE from *http://java.sun.com/javase/downloads/index.jsp*.

Note: Video recording is only supported on Windows client machines.

Recording Videos in the Web UI

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the manual execution plan that you want to execute.
3. Click Run. The **Execution Plan Run Details** page displays in the **Current Run** tab.
4. In the **Assigned Tests** grid, select the **Start Video Recording** button from the **Actions** column. The **SCTM Video Recording Web Client** application opens and begins capturing video of the screen.
5. Run your manual test.
6. When your test is complete, click **Stop Video Recording** on the **SCTM Video Recording Web Client**. The **SCTM Video Recording Web Client** displays new fields for saving the recording.
7. Rename your video recording in the **Result filename** field.
8. Click **Upload File** to save the recording into the manual test. You will receive a confirmation message when the file is successfully uploaded. The file can be found as a **Result File** in the **Test Details** area of the **Execution Plan Run Details** page.
9. If you want to save the file locally, click **Save File**.

Manual Testing Client

The Manual Testing Client is an Eclipse-based client tool that provides the full functionality you need to efficiently execute manual tests. The client enables testers to manage their tests, edit tests, and track results without the need of an Internet connection.

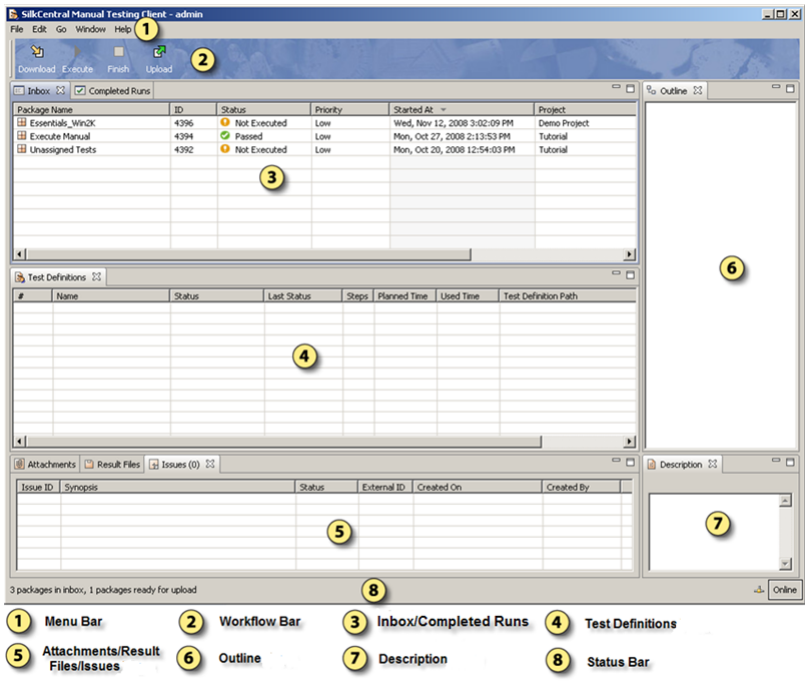
Tip: If many users work on manual tests in parallel, we recommend to use the Manual Testing Client, an Eclipse-based client tool, to execute the manual tests.

Note: The Manual Testing Client supports running test executions with code analysis information. If a test execution is already enabled to gather code analysis information within Test Manager, the settings are automatically available in the Manual Testing Client. Code analysis can also be enabled for execution plans from within the Manual Testing Client.

Tour of the Manual Testing Client UI

Note: Eclipse does not initialize views until they are activated for the first time, so certain tabs within the Manual Testing Client will not display their contents until they are selected. For example, the **Attachments** page will not display the number of attachments it tracks until it is selected.

The following image displays the eight main areas of the UI of the Manual Testing Client.



Menu Bar

The menu bar provides editing options for tests and views. You can use the menu bar to edit user, appearance, or test settings, to navigate between execution packages, or to access **Help**. The following list displays the available menus:

Menu	Description
File	Provides the same functionality as the workflow bar and additional functionality for package import and export, changing users, switching the Manual Testing Client between online and offline mode, and storing executed package results to Test Manager.
Edit	The edit menu includes the following items:

Item	Description
Set as <Status>	Set the execution status of the execution package or test to the selected status.
Add Result File...	Add a result file to an execution package or test.
Edit Build Number...	Select the build number for an execution package.
Edit Code Analysis Settings...	Specify hostnames to include in code coverage runs for an execution package. The hostnames must be separated by commas. For example: labmachine1, 192.168.0.1:19129.
Delete	Delete an execution package. You can delete only execution packages with completed runs.

Go	Navigate between the available execution packages and tests.
Window	Edit the view and the preferences of the Manual Testing Client. The menu includes the following items:

Menu	Description

Item	Description
Show View	Activate and deactivate the available views in Manual Testing Client's **Workspace**.
Reset Perspective...	Reset the current Manual Testing Client perspective to the default state.
Preferences...	Set the preferences for the Manual Testing Client.

Help	Access the documentation and **About** page of the Manual Testing Client.

Workflow Bar

The workflow bar gives you quick access to the basic functions you can perform with the Manual Testing Client. The following buttons are available in the workflow bar:

Button	Description
Download	Download the execution packages from Test Manager.
Execute	Execute a test in the Manual Testing Client.
Finish	Stop the execution of a test before the test is fully executed.
Upload	Upload the execution packages to Test Manager.

Inbox

The **Inbox** lists all execution packages that have been downloaded to the Manual Testing Client for manual execution. Multiple execution packages can be selected within the **Inbox** window using standard Windows keyboard shortcuts. The **Inbox** includes the following properties for the selected test:

Property	Description
Package Name	Name of the execution package that has been downloaded from Test Manager.
ID	Execution package number that has been generated for this execution package.
Status	Status of the execution package. Status values cannot be edited on the **Completed Runs** page. The available values are `Not Executed`, `Passed`, `Failed`, `Unresolved`, and `Unsupported`. Execution package status values cannot be edited. The overall package status is determined by the composition of the test statuses, and by extension the statuses of the test steps, that are contained within a package. For example, overall package status remains `Not Executed` as long as a test step of one of the contained tests has a status of `Failed`. So, even if some of a test's steps have passed, the overall status of the package will remain `Not Executed` until the package is finished and all not executed tests are assigned a status on the **Finish Run** dialog box. The overall status will be considered `Passed` when one or more test steps or tests in a package are `Passed` and all not executed steps and tests are resolved through the **Finish Run** dialog box.
Priority	Priority of the execution package.
Keywords	All keywords that are assigned to the execution package.
Started At	When testing of the execution package began.

Property	Description
Project	Name of the Test Manager project from which the execution package was derived.
Version	Product version from which the execution package was derived.
Build	Product build from which the execution package was derived.
Execution Path	File path where this execution plan resides in Test Manager's **Execution Plans** tree.

Completed Runs

The **Completed Runs** page lists all execution packages for which testing is complete. The **Completed Runs** page includes the following properties for the selected execution package:

Property	Description
!	This column offers a status icon that indicates the upload status of corresponding execution packages. A red arrow icon indicates that a package's results have not yet been uploaded. A checkmark icon over a faint arrow icon indicates that a package's results have already been uploaded to the server. Double-click this icon to open the **Execute Test** dialog for the first test of this execution package.
Package Name	Name of the execution package that has been downloaded from Test Manager.
ID	Execution package number that has been generated for this execution package.
Status	Status of the execution package. Status values cannot be edited on the **Completed Runs** page. The available values are `Not Executed`, `Passed`, `Failed`, `Unresolved`, and `Unsupported`. Execution package status values cannot be edited. The overall package status is determined by the composition of the test statuses, and by extension the statuses of the test steps, that are contained within a package. For example, overall package status remains `Not Executed` as long as a test step of one of the contained tests has a status of `Failed`. So, even if some of a test's steps have passed, the overall status of the package will remain `Not Executed` until the package is finished and all not executed tests are assigned a status on the **Finish Run** dialog box. The overall status will be considered `Passed` when one or more test steps or tests in a package are `Passed` and all not executed steps and tests are resolved through the **Finish Run** dialog box.
Priority	Priority of the execution package.
Keywords	All keywords that are assigned to the execution package.
Started At	When testing of the execution package began.
Finished At	When testing of the execution package ended.
Project	Name of the Test Manager project from which the execution package was derived.
Version	Product version from which the execution package was derived.
Build	Product build from which the execution package was derived.
Execution Path	File path where this execution plan resides in Test Manager's **Execution Plans** tree.

Tests

The **Tests** page includes all information related to the manual test that is selected above in the **Inbox** or **Completed Runs** page.

Multiple tests can be selected within the **Tests** window using standard Windows keyboard shortcuts. To apply a status change to selected tests, right-click the selection and select a new status value from the context menu.

The **Tests** page includes the following properties for each test:

Property	Description
#	Number that has been automatically generated for the test.
Name	The test name.
Status	Status of the test. The available values are `Not Executed`, `Passed`, `Failed`, `Unresolved`, and `Unsupported`. This value can be changed by right-clicking the current value and selecting an alternative value from the context menu.
Last Status	Status that this test held before the current status.
Steps	Number of steps in the selected manual test.
Planned Time	Estimated time for completion of the test in `[hh:mm:ss]`.
Used Time	This field tracks the elapsed time in `[hh:mm:ss]` since the start of the test execution. This field can be manually edited. The timer will stop during editing. After editing this field the timer will continue tracking time from the manually adjusted time.
Test Path	File path where this test resides in Test Manager's **Tests** tree.

Attachments

The **Attachments** page lists any attachments related to the selected manual test. This page is also available on the **Execute Test** dialog box.

When you have selected a test in the **Tests** window, you have the option of supplementing the list of displayed attachments by selecting an **Include attachments of** value. Select **Test Container/Folders** to include all attachments from the selected test's test container or folder. Or select **Test Steps** to include attachments from the test steps of the test.

The **Attachments** page includes the following properties for each attachment:

Property	Description
Name	Name of the attachment.
Type	Attachment file type.
Description	Description that has been created for the attachment, if any.
Source	File path where this attachment's test resides in Test Manager's **Tests** tree.
Image Preview	If the attachment is an image, you can use the **Image Preview** controls to view the attachment. Right-click the image, or click the buttons to the right of the window, to access the following commands: **Show Actual Size**, **Scale to Fit**, and **Scale to Fit — Keep Aspect Ratio**. Click **Open as Detached Window** to open **Image Preview** in a separate window.

Result Files

The **Result Files** page lists any result files that are related to the selected manual test. This page is also available on the **Execute Test** dialog box. The **Result Files** page includes the following properties for each result file:

Property	Description
Name	Name of the result file.
Source	File path where this result file's test resides in Test Manager's **Tests** tree.
Add File	Click to browse to and select a new result file for upload to this test.

Property	Description
Paste Image	Click to paste an image from your computer's clipboard and attach the image to this test.
Remove	Click to remove the selected result file attachment from this test.
Image Preview	If the result file is an image, you can use the **Image Preview** controls to view the result file. Right-click the image, or use the buttons to the right of the window, to access the following commands: **Show Actual Size**, **Scale to Fit**, and **Scale to Fit — Keep Aspect Ratio**. Click **Open as Detached Window** to open **Image Preview** in a separate window.

Issues

The **Issues** page lists any issues related to the selected manual test. This page is also available on the **Execute Test** dialog box. The **Issues** page includes the following properties of each issue:

Issue ID	ID that has been assigned to this issue.
Synopsis	Synopsis that has been written for this issue.
Status	Status of the issue.
External ID	Indicates if the issue is tracked by an external issue tracking system. If this issue is tracked by an external issue tracking system, and that issue has been assigned an ID, you can click the external ID number in this field to link directly to the issue in the external issue tracking system.
Created On	When the issue was created.
Created By	User who created the issue.

Outline

Shows the content tree of the selected execution package or the location of the selected test in the execution package.

Description

Shows the description of the selected execution package or test.

Status Bar

Shows the current amount and status of the execution packages and tests in the currently active view.

Button	Description
Online/Offline	Click to switch Manual Testing Client's mode from online to offline and back.

Installing Manual Testing Client

The Manual Testing Client is a separate executable from Test Manager. Using Java Web Start technology, the Manual Testing Client can be deployed with a single click over the network. Java Web Start ensures the most current version of Manual Testing Client is deployed, as well as the correct version of the Java Runtime Environment (JRE).

Prerequisite: Java Runtime Environment (JRE) version 6 or higher must be installed on your computer to use Manual Testing Client with Java Web Start technology. You can download JRE from *http://java.sun.com/javase/downloads/index.jsp*.

Alternately you can install Manual Testing Client on your computer by navigating to **Help** > **Tools** > **Manual Testing Client** .

Tip: To start the Manual Testing Client, navigate to **Start** > **Programs** > **Silk** > **SilkCentral** > **Test Manager 12.0** > **Manual Testing Client** .

To install and start Manual Testing Client:

1. When clicking a link to the Manual Testing Client Web Start URL, `http://<Test Manager host>:<port>/webstart/mtc/`), for example in a manual testing notification email, a **File Download** dialog box opens.
2. Click **Open**.

   If Manual Testing Client is not installed on your computer, the **Java Web Start** dialog box opens and immediately starts downloading Manual Testing Client. The download can take up to several minutes. If Manual Testing Client is already installed on your computer, Manual Testing Client opens and the following steps are not applicable.

   When the download has completed, a **Warning — Security** dialog box opens, asking you if you want to run the digitally signed application.
3. Check the **Always trust content from this publisher** check box, then click **Run**. The Manual Testing Client opens.

Uninstalling Manual Testing Client

To uninstall the Manual Testing Client:

1. Open a command line window.
2. In the Windows command line window, type `javaws –viewer` and press **Enter**. The **Java Cache Viewer** opens.
3. In the **Show** list box, select `Applications`, if not already selected.
4. The table should now list Manual Testing Client — Web Start. Select this application and click the delete button (X) on the toolbar.
5. Manual Testing Client is now removed from your computer and you can close the **Java Cache Viewer** and **Java Control Panel** dialog boxes.

Tests in the Manual Testing Client

While in Edit mode, the Manual Testing Client offers a full range of test-editing functionality, including the addition, reordering, and removal of test steps and the insertion of step properties. You can edit manual test properties in both online and offline modes. You can synchronize changes made in offline mode with the server whenever an Internet connection is available.

Edit mode enables editing of all test elements that can be edited through the Test Manager Web client. On the **Details** page this includes the following values:

- **Planned Time**
- **Step Names**
- **Step Property Values**
- **Step Description**
- **Expected Result**

On the **Description** page the **Test Description** field can be edited, including the insertion of step property parameters.

In Edit mode, multiple test steps can be selected within the **Tests** window using standard Windows keyboard shortcuts. To apply a status change to selected test steps, right-click the selection and select a new status value.

In normal mode, you can only enter test results into the **Result** column and edit the status of tests and individual test steps. Statuses are changed by right-clicking status values and selecting an alternative status value.

Note: Editing of data-driven tests is not supported.

Parameters

Step property parameters can be inserted into test and test step descriptions. Parameters can be inserted into the **Test Description**, **Step Description**, and **Expected Result** fields. In normal mode, parameter values are resolved (their parsed values are displayed in place of the parameters themselves). In Edit mode parameters are not resolved; the parameters themselves are displayed. When in Edit mode, using the **Parameters** list box on the **Description** tab toolbar, you can select pre-configured Test Manager parameters for insertion.

Change Conflict Handling

With the Manual Testing Client's offline editing functionality it's possible for multiple users to edit the same test at the same time. Test Manager automatically merges all uploaded changes into the **Tests** tree unless change conflicts arise, for example if two users simultaneously edit the same manual test step. If your uploaded changes conflict with recent changes made by another user, upon upload, you will be presented with the **Test Conflicts** dialog box so that you can manage the conflict.

Using the **Test Conflicts** dialog box you can specify if your changes should be saved as a part of the test and included in future runs of the test, or if they should be ignored. If you opt to have your changes ignored, they will still remain a part of the test results from the test run in which they were included.

Inserting Test Manager Parameters into Description Text Boxes

To insert a Test Manager parameter into a description text box:

1. In the Manual Testing Client, in Edit mode, navigate to **Execute Test** > **Description** or **Execute Test** > **Details** .

   You can insert any pre-configured Test Manager parameter into the **Test Description**, **Step Description**, and **Expected Result** text boxes.
2. Place your cursor into the text box.
3. Click **Parameters** on the toolbar.
4. Select a pre-configured Test Manager parameter from the list box.

Configuring the Manual Testing Client

This section explains how to configure the Manual Testing Client.

Configuring Connection Parameters

Connection parameters are configured automatically the first time you start the Manual Testing Client.

To edit login credentials and validate your connection:

1. In the Manual Testing Client, navigate to **Window** > **Preferences** . The **Preference Settings** dialog box opens.
2. Type the URL of your Test Manager installation in the **Test Manager server URL** text box.
3. Select **Remember Credentials** if you want the Manual Testing Client to insert your login credentials automatically the next time you start the application.
4. Enter your Test Manager **Username** and **Password**.
5. Click **Validate Connection** to test your login settings.
6. Click **OK** on the confirmation dialog box.
7. Click **OK** to save your settings.

Configuring Other Settings

To configure other settings:

1. In the Manual Testing Client, navigate to **Window** > **Preferences** . The **Preference Settings** dialog box opens.
2. In the **Other settings** section, check or uncheck one or more of the following check boxes:

Download attachments automatically	Download attached files automatically when execution plan packages are downloaded from Test Manager. This option must be enabled if you intend to work offline after you download your assigned execution packages.
Ask for build number when completing packages	Before uploading packages to Test Manager, display a prompt requesting the build number on which the test was performed.
Show execution dialog always on top	Have the **Execute Test** dialog box display on top of other open windows on your computer desktop to facilitate manual testing. When enabled, the **Execute Test** dialog box stays on top even when another window has the focus. When executing manual tests, you may want to keep the **Execute Test** dialog box on top so that you can easily enter your test results. If your computer monitor is too small to contain both the **Execute Test** dialog box and the application under test, you should leave this setting unchecked.
Ask for uploading workspace to SilkCentral Test Manager before closing main window	Display a confirmation prompt before closing the Manual Testing Client.

3. Click **OK** to save your preference settings.

Configuring Package Upload Preferences

To configure execution-package upload preferences:

1. In the Manual Testing Client, navigate to **Window** > **Preferences** . The **Preference Settings** dialog box opens.
2. In the **Packages** area of the dialog box, check the **Remove uploaded packages** check box to define an option for automatic deletion of execution plan packages from the Manual Testing Client after packages are uploaded to Test Manager.
3. Click one of the following deletion option buttons:
   - **Immediately**
   - **After <x> days**. Type a value in the days text box if you select this option.
4. Check the **Upload completed packages immediately** check box if you want to have completed test run packages uploaded to Test Manager automatically after test runs are completed.

Managing Attachments with the Manual Testing Client

This section explains how to manage attachments with the Manual Testing Client.

Pasting Screen Captures

To upload a screen-captured image directly from the clipboard:

1. Copy a screen capture to your computer's clipboard.
   The **Paste Image** button on the **Result Files** page of the Manual Testing Client becomes enabled.
2. Click **Paste Image**.

3. Specify a **File Name** for the image on the **Paste From Clipboard** dialog box.
4. Click **OK** to save the copied screen capture as an image file attachment.

Uploading Attachments to the Manual Testing Client

When during the course of testing you encounter a relevant result file, which can be a screen-captured image, an error log, or another file, you can upload the file as an attachment to a test.

To upload a result file as an attachment:

1. In the Manual Testing Client, select a test in the **Tests** page.
2. Click the **Result Files** tab.
3. Click **Add File** to browse to and select the result file you want to upload.
4. Click **Open** to attach the file.

Viewing Attached Images Within the Manual Testing Client

To view an attached image file:

1. In the Manual Testing Client, select an image file in the **Attachments** page. The image displays in the **Image Preview** field.
2. Use the following viewing tools next to the **Image Preview** field to manipulate the image:
   - Show Actual Size
   - Scale to Fit
   - Scale to Fit - Keep Aspect Ratio
   - Open as Detached Window

Viewing Attachments Within the Manual Testing Client

To view a test attachment from within the Manual Testing Client:

1. In the Manual Testing Client, select a test in the **Tests** page.
2. The **Attachments** page lists all of the result files that are associated with the selected test.
3. Using the **Test Container/Folders** and **Test Steps** check boxes, you can filter the list of attachments to include only those attachments that are related to the selected test container, test folder, or test step.

Working with Execution Packages

Downloading Execution Packages

After the Manual Testing Client is configured, the first step in running manual tests is downloading copies of the manual execution plans that are assigned to you.

Note: Execution plans remain online in Test Manager; only copies of the execution plans are downloaded to the Manual Testing Client.

To download the execution packages that are assigned to you:

1. In the Manual Testing Client, click **Download** on the toolbar.
2. If your connection settings are correctly configured, and execution plans are started in Test Manager, the execution packages that are assigned to you display in the **Inbox**.

Exporting Execution Packages

Downloaded execution packages can be both exported from and imported to the Manual Testing Client. This allows for easy exchange of execution packages between testers over email. Downloaded attachments are automatically included in exported packages.

Note: Execution packages have the `.zpkg` extension.

To export an execution package:

1. In the Manual Testing Client, right-click an execution package in the **Inbox** page.
2. Select **Export Package**.
3. On the **Export to** dialog box, browse to the location where the package is to be saved and click **Save**.
4. Click **OK** on the confirmation dialog box notifying you that the export was successful.

Importing Execution Packages

Downloaded execution packages can be both exported from and imported to the Manual Testing Client. This allows for easy exchange of execution packages between testers over email. Downloaded attachments are automatically included in exported packages.

Note: Execution packages have the `.zpkg` extension.

To import an execution package:

1. In the Manual Testing Client, navigate to **File** > **Import Package** .
2. On the **Import from** dialog box, browse to the package that is to be imported and click **Open**.
3. Click **OK** on the confirmation dialog box notifying you that the import was successful.

Editing Package Build Numbers

To edit the build number of an execution package:

1. In the Manual Testing Client, right-click an execution package in the **Inbox** page.
2. Select **Edit Build Number**. The **Select Build Number** dialog box opens.
3. Select a build number from the **Build** list box.

   Note: You can refresh the build list by clicking **Refresh build list**.

   Note: If you want to be prompted to specify a build number each time a test run is completed, check the **Ask for build number when completing packages** check box.
4. Click **OK**.

Uploading Test Results to Test Manager

To upload test results from finished packages:

1. In the Manual Testing Client, complete a manual test by clicking **Finish Run** on the **Execute Test** dialog box or **Finish** on the toolbar.

   Note: If you attempt to complete testing of a test package while any of the package's tests have a status of `Not Executed`, the **Finish Run** dialog box will display on which you can define how not executed tests should be handled.
2. Click the **Completed Runs** tab.
3. Right-click a completed test run and select **Upload to Test Manager**, or select **Upload** from the **File** menu, to upload your test results to Test Manager.

   Note: Execution plan statuses are updated automatically in the SilkCentral database when you are working online.

Managing Change Conflicts upon Upload

To manage change conflicts during upload:

1. After completing your manual test edits, click **Upload** to upload your results to the server.
2. Click **Yes** to confirm that you want to have your changes committed to the **Tests** tree on the server. If your changes conflict with recent changes made by another user, the **Test Conflicts** dialog box will display, listing the tests that are in conflict.
3. To directly access a conflicting test in Test Manager and view the changes, right-click the test and select **Go to Test in Test Manager**.
4. Click **Upload Changes** to ignore changes made by other users and commit your changes to Test Manager, thereby overwriting any recent changes that conflict with your changes, or click **Revert Changes** to not have your changes saved to the test.

   If you opt for **Revert Changes**, your changes will not be committed to the **Tests** tree, however your changes will be visible in the execution results you are uploading. Your changes will not be included in future runs of the test.

Storing Test Results to Test Manager

To store test results from Manual Testing Client in Test Manager:

1. In the Manual Testing Client, select **Store Results to SilkCentral** from the **File** menu.
2. Alternatively, when closing Manual Testing Client, you can upload your entire workspace to Test Manager.

Adding an Internal Issue with the Manual Testing Client

The Manual Testing Client enables you to create an internal issue and map an external issue to this internal issue. To create the internal issue and map the external issue:

1. In the Manual Testing Client, click the **Inbox** tab.
2. Select an execution plan package.

   The tests included in the selected package are listed in the **Tests** page.
3. In the **Tests** page, double-click a test.
4. Click **New Internal Issue**. The **New Issue** dialog box opens.
5. Fill out the text boxes.
6. Click **OK**.

Note: You must be online during this procedure.

Enabling Code Analysis Within the Manual Testing Client

To enable code analysis for an execution plan from within the Manual Testing Client:

1. In the Manual Testing Client, navigate to **Edit** > **Edit Code Analysis Settings** . The **Edit Code Analysis Settings** dialog box opens.
2. Check the **Enable code analysis for package <Package Name>** check box.

   Note: After code analysis is enabled, you can execute your tests in the Manual Testing Client. However, you need to click **Code Analysis: Start** on the **Details** page of the **Execute Test** dialog box before you actually start testing. This way Test Manager will collect code analysis information while you execute the manual test. When you are done testing, click **Stop** to halt the collection of code analysis information.
3. Select a profile from the **Code Analysis Profile** list box.
4. In the **Hostnames** text box, enter a comma-separated list of hostnames, with port, if default port 19129 is not used, from which code analysis information is to be gathered.

   For example `labmachine1, labmachine2:8000, 198.68.0.1`. For each execution plan, you need to define the host names of the machine resources where the AUT is running. For example, with a

client/server system, you must not only gather code coverage information on the client, which probably runs directly on an execution server, but also on the server, which likely runs on a different machine. This applies to all multi-tiered applications.

Note: For JUnit code analysis runs, you do not need to specify a hostname.

5. Click **OK** to save your settings.

Working Offline with the Manual Testing Client

If you plan to work without access to an Internet connection, select offline work mode so that the Manual Testing Client will not attempt to connect to Test Manager automatically.

To work in offline mode:

1. In the Manual Testing Client, select **File** > **Work Offline** or click **Online** in the bottom right corner.
2. Once you have completed your tests and have access to an Internet connection, proceed with uploading your test results.

Execute Test Dialog Box

When you execute a manual execution plan in the Manual Testing Client, the **Execute Test** dialog box displays. This dialog box enables you to execute the test steps that are included in the selected execution plan, and track the results of the test steps.

The **Execute Test** dialog box includes the following pages: **Description**, **Details**, **Attachments**, **Result Files**, and **Issues**.

Description

In normal mode, the **Description** page offers a read-only **Test Description** text box for the selected test and the **Test Name** text box. In edit mode, these text boxes can be edited. Click **Open Description in Separate Window** to open the most recently saved version of the test description in a separate window. The detached window always remains on top to assist in manual testing.

Details

The **Details** page offers a read-only description of the selected test.

The page includes the following items:

Item	Description
Test Status	Current status of the test. In edit mode this text box can be edited.
Last Status	Status of the test in the previous test run.
Planned Time	Estimated time for completion of the test in `hh:mm:ss`. In edit mode this text box can be edited.
Used Time	This text box tracks elapsed time in `hh:mm:ss` since the beginning of test execution. This text box can be manually edited. The timer stops during editing. After editing this text box the timer continues tracking from the manually adjusted time.
Start	Initiates code analysis. Enter the host names for which you want to run code analysis.
Test Steps	Lists the manual test steps the selected test includes. You can select multiple test steps in the **Test Steps**

Item	Description
	section by using the standard Windows keyboard shortcuts. To change the status of the selected test steps, right-click on the steps and choose a new status value.
Result	The result of each test step as observed by the tester. Edit this text box after you have completed this step.
Status	Status of each step. Edit this text box after you have completed this step.
Step Description	Describes the action you must perform in this step.
Expected Result	The expected result or success condition of each test step.

Execute Test Toolbars

The **Execute Test** dialog box includes the following toolbars:

Toolbar **Description**

Test Steps Use to manage your manual test steps during a test execution. The toolbar includes the following items:

Item	Description
Add Test Step	Add a new test step to the end of the **Test Steps** list.
Insert Test Step	Insert a new test step above the selected test step in the **Test Steps** list.
Duplicate Test Step	Create a copy of the selected test step in the **Test Steps** list.
Delete Test Step	Delete the selected test step from the **Test Steps** list.
Move Test Step Up	Move the selected test step up one position in the **Test Steps** list.
Move Test Step Down	Move the selected test step down one position in the **Test Steps** list.

Text Formatting Use to format the **Test Description**, **Step Description**, and **Expected Result** descriptions. Click **Parameters** to insert parameters into your descriptions. The toolbar includes the following items:

Item	Description
Bold	Apply bold formatting to the selected text.
Italics	Apply italicized formatting to the selected text.
Underline	Apply underlined formatting to the selected text.
Align Left	Align the selected text to the left side.
Align Center	Align the selected text to the center.
Align Right	Align the selected text to the right side.
Justify	Apply a justified alignment to the selected text.

Toolbar **Description**

Item	Description
Bulleted List	Convert the selected text to a bulleted list.
Indent Left	Apply a left-side indent to the selected text.
Indent Right	Apply a right-side indent to the selected text.
Undo Change	Undo the last action you performed in a text description text box.
Redo Change	Redo the last action you performed in a text description text box.
Font	Apply a different font type to the selected text.
Font Size	Apply a different font size to the selected text.
Format	Apply a different pre-defined formatting style to the selected text. For example, Heading 1, Heading 2, ...
Parameters	Insert pre-configured Test Manager custom step properties, which are also called project parameters, into text descriptions. In normal mode, the dialog box displays the parsed values of the resolved parameters. In Edit mode, the dialog box displays the actual parameters.

Navigation Use to manage your manual test execution and to navigate between tests. The toolbar includes the following items:

Item	Description
Edit	Edit the properties of the selected test.
New Internal Issue	Add an internal issue to the selected test.
Go To Issues	View the issues of the selected manual test in Test Manager, or assign new issues to the test.
Previous Test	Return to the previous test step in the manual test execution.
Next Test	Advance to the next test step in the manual test execution.
Finish Run	Close the **Execute Test** dialog box when you have completed all test steps in the active manual test.

Video Capturing in the Manual Testing Client

You can capture a video of a test in Manual Testing Client, which you can then add to the Manual Testing Client result files. You can then analyze the video, for example if an error occurs during the test.

Note: Video capturing is currently enabled only for Microsoft Windows operating systems. To use video capturing on a Windows 2008 system, you need to install the following server features:

- Desktop Experience

- Quality Windows Audio Video Experience

Note: The main display of your system is recorded. Secondary displays are not included in the video. During the recording, some tasks are not enabled. To perform such tasks, stop the recording.

Recording Videos in the Manual Testing Client

To enable video capturing in the Manual Testing Client:

1. In the Manual Testing Client, click on the package that includes the test that you want to execute in the **Inbox**. The tests in the selected package are displayed in the **Tests** page.
2. Double-click on the test that you want to execute. The **Execute Test** dialog box opens.
3. Click the **Details** tab. The **Details** page opens.
4. In the **Test Details** section, click **Start** on **Video Recording**. A red button indicates that a video of the test is being recorded.
5. When you no longer want to record, click **Stop** on **Video Recording**. The **Add Video to Result Files** dialog box opens.
6. Type a name for the video file, and click **OK**, to save the video, or click **Cancel** to delete the video file. The video is added to the result files of the test.
7. To view the video, click the **Result Files** tab and double click on the video file.

Analyzing Test Runs

This section describes how you can analyze test runs with Test Manager.

Changing the Status of a Test Run

To manually change the status of a test run:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Runs** tab.
4. Select the execution plan run. The test section of the **Runs** page lists the test runs.
5. Click on the **Run ID** of the test.
   The **Test Run Results** dialog box displays.
6. Click the **Details** tab.
7. Click **Change Status**. The **Change Status** dialog box opens.
8. Select the new status for the test run from the **New Status** list box.
9. Type an explanation for the manual status change in the **Comment** text box.

   Note: Inserting a comment is mandatory.

10. Click **OK** to confirm the status change.

Note: Status changes produce history changes. To view the history of all status changes for the test run, click the **Messages** tab in the **Test Run Results** dialog box.

Viewing Test Run Details

To view the details of a test run:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan in the **Execution** tree.
3. Click the **Runs** tab.

4. Click the **Run ID** of the execution for which you want to see details. The **Test Run Results** dialog box displays.
5. Click the **Details** tab.

Deleting Test Run Results

To delete the results of a specific test run:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution Plans** tree, select the execution plan that you want to edit.
3. Click the **Runs** tab.
4. In the **Actions** column of the test run for which you want to delete results, click ✕.
5. Click **Yes** on the subsequent confirmation dialog box to complete the deletion.

Deleting the Results of an Execution Plan

To delete some or all of the run results of an execution plan, a folder, or a project:

1. In the menu, click **Execution Planning** > **Details View**.
2. Right click on the element in the **Execution** tree and select **Delete Results**.
3. The **Delete Results** dialog box displays. Select which results you want to delete.

Option	Description
All runs except the last run	Deletes the results of all execution plan runs except the results of the most recent run.
All runs within the time span	Deletes the results of all execution plan runs in the specified time span. With this option selected, click next to the **From** and **To** fields to specify dates using the calendar tool.

4. Click **OK** to confirm the deletion.

Note: Results of tagged builds cannot be deleted. To delete these results, untag the build in **Administration** > **Products, Versions, and Builds** > **Products**. For more information, refer to the *SilkCentral Administration Module Help*.

Execution Plan Run Results

Tracking > **Activities** > **Last Executions** > **Run ID**

The **Execution Plan Run Results** dialog box lists the details of an execution plan run.

You can also access the dialog box from **Execution Planning** > **Details View**. Select the execution plan for which you want to see details, click the **Runs** tab, right click on the run and choose **View details**.

The dialog box shows detailed information about the following items:

Item	Description
Execution Plan Name	Name of the execution plan.
Execution Plan ID	Unique identifier of the execution plan.
Execution Plan Run ID	Identifier of the execution plan run.
Start Time	Time the run was started.
Duration	Time consumed to execute all included tests. Includes the duration of the setup and cleanup test and the time consumed to fetch automation files and code coverage,

Item	Description
	launch execution tools, and other tasks. For manual executions this is the time between starting and finishing the execution.
Execution Server	Execution server assigned to the execution plan.
Warnings/Errors	Amount of warnings and errors generated during the run.
Status	Displays the number of passed, failed and not executed tests in a bar graph.
Version/Build	Version and build of the product specified for the run. These either are set by the user for this execution plan, or are read from the build information file at execution time. If the build is tagged, it is marked bold and with .
SilkTest Classic AUT Host Name	Name of the SilkTest Classic AUT (Agent Under Test) Host.
Setup Test	The test that prepared the testing environment in anticipation of the test. Click on the name of the test to view or edit it. Click on the ID of the test run to open the **Test Run Results** dialog box.
Cleanup Test	The test that restored the testing environment to its original state following the test. Click on the name of the test to view or edit it. Click on the ID of the test run to open the **Test Run Results** dialog box.

The **Execution Plan Run Results** dialog box provides additional information about the files included and the messages generated during the execution plan run. It also lists all the assigned tests for the execution plan.

For manual tests click **Manual Test Results** to get a read-only version of the **Current Runs** page, with detailed information on the manual test.

The **Assigned Tests** section lists all tests that are assigned to this execution plan. Click on the name of a test to view or edit it, or click on the **Run ID** of a test to open the **Test Run Results** dialog box.

Test Run Results

Execution Planning > **Details View** > **Runs** > **<Execution Plan>** > **Assigned Tests** > **<Run ID>**

The **Test Run Results** dialog box lists run details of a test. You can access the dialog box from the following locations:

- **Tests** > **Details View** > **<Test>** > **Runs** > **<Run ID>**
- **Execution Planning** > **Details View** > **Runs** > **<Execution Plan>** > **Assigned Tests** > **<Run ID>**
- **Tracking** > **Activities** > **Last Executions** > **<Run ID>** > **Assigned Tests** > **<Run ID>**

The **Test Run Results** dialog box displays the following tabs:

Tab	Description
Details	Shows the details of the test run, including its **Duration**, **Execution Path**, the **Execution Plan Run ID** of the execution plan run that included the test run, any **Warnings/Errors**, and the run comment, if a comment is added to the run. This tab also allows you to change the status of the test run. This option is useful if you need to manually overrule the status of a test run.

Tab	Description
	When a manual status change is performed, the details of the change are reflected in this tab's **Status**, **Status Changed On**, **Status Changed By**, **Previous Status**, and **Status Change Comment** fields.
Timeline	The **Timeline** tab shows a grid that allows easy result and error analysis of test runs by combining the following: • The structure of the test suite if an `output.xml` file is available including failure/error/warning count. • Incidents from the `output.xml` file. • Log entries and messages. • Result files. The entries are listed chronologically and the view contains check boxes allowing you to filter the different types of items. You can also access this tab from the **Analyze Results** and **Analyse Errors** actions in the **Test Run** grids. **Note:** The tab is displayed for all test types except for manual tests and SilkTest Workbench tests.
Specific	Only displayed for SilkTest Classic, SilkPerformer, and manual tests. This tab includes details that are specific to the selected test type. For example, when a SilkTest Classic test is selected, this view includes the selected test case, test data, and any warnings that were displayed during the test run.
Files	Lists all files that were generated by this test run, along with file sizes. The names of SilkTest Classic .rex files act as download links. Once downloaded, these files can be viewed directly in a text editor. The upper table lists files that are associated with the test, such as result files or manually uploaded files for manual tests. The lower table lists files that are associated with the execution plan, for example execution log files or code analysis results. Click **Download All Files** to download all result files generated by the test run, as a zipped package. For SilkTest Workbench, click the `result.stwx` file to open SilkTest Workbench in context of the result file. For SilkTest Workbench, click the `error.png` to see the last screenshot for a Visual Test when a playback error occurs.
Messages	Lists all messages that were generated by this test run, along with the severity of the messages. Messages that are associated with an execution plan as a whole, and not to one of the individual tests, can be viewed in **Executions** > **Activities** > **Messages**
Success Conditions	Only displayed for automated tests. This tab shows all the success conditions that are defined for the test in **Tests** > **Details View** > **<Test>** > **Properties** during the test planning process, and the result values from the execution run. Success conditions are used to determine if a test is `Passed` or if it has `Failed`.
Data Driven	Only displayed for data-driven tests using the option of having a single test for all data rows of the data set. This tab lists the status of each instance (data row) run of the test. Clicking an instance brings up another instance of the **Test Run Results** dialog box with run details of the selected instance.

Tab	Description
Attributes	Any attributes that are configured for the test.
Parameters	Any parameters that are configured for the test.

The following table lists the UI elements that are used to step through the test results of an execution run. These elements are only visible when accessing the **Test Run Results** dialog box from an execution plan.

Item	Description
Skip Passed	Used to determine which test run results should be displayed when browsing using the **Previous Result** and **Next Result** buttons. Checking this option only displays tests with a status other than `Passed`.
< Previous Result	Jumps to the result details of the previous test in the selected execution plan run.
Next Result >	Jumps to the result details of the next test in the selected execution plan run.

Viewing Execution Activities for Data-Driven Tests

To view execution activities for data-driven tests:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select an execution plan that is based on a data-driven test.
3. In the menu, click **Tracking** > **Activities**.
4. Click the **Run ID** of the relevant execution plan.
5. In the **Assigned Tests** table, click the **Run ID** of a data-driven test.

   Note: If you are running a multiple data-driven test, you will see one test for each data row in your data source.

   The results page for that particular test opens.
6. Click the **Data Driven** tab. Here you can view all instances of the test that were executed.

   Note: The test's data-driven properties are listed on the **Details** page in the **Data-driven Properties** table.
7. Click an instance name to view test run details for that specific instance.

   Note: If you are working with multiple data-driven test instances, a separate instance will be created for each data row in your data source.
8. Click the **Parameters** tab to view the data source values that were used during this specific test run.

Execution Runs Page

Execution Planning > **Details View** > **<Execution Plan>** > **Runs**

The **Runs** page shows statistics regarding all the runs of the selected execution plan.

For configuration suites, the page displays all runs of the included configurations.

For testing cycles, the page displays all runs of the included execution plans.

The page is split into two separate sections, one listing the execution plan runs, and the second listing the test runs for the execution plan run selected in the first section.

The grid views on the **Runs** page offer view settings, including resizing and reordering of columns, filtering, sorting, and grouping options that are configurable on a per-user basis. You can display or hide columns,

adjust the width of columns, and move columns around by clicking on a column and dragging it to the desired location. You can also use the keyboard to navigate through the runs.

To open the **Execution Plan Run Results** dialog box, right-click on a run and choose **View Details**. To compare two execution plan runs, use **Ctrl + Click** or **Shift + Click** to select the two runs. Right-click on your selection and select **Reports** > **Execution Plan Run Comparison ...** .

The execution plan runs section lists the runs of the selected execution plan. The section is paged with fifty runs shown on a page. Use the arrow buttons to navigate through the pages. The page shows the following columns for each run:

Column	Description	
Actions	Actions that you can perform on the execution plan.	
	Delete Run Results	Click to delete the results of this run. When you delete the results for selected runs, Test Manager removes the runs from the **Runs** page. The runs are grayed out until the background process completes the deletion. Alternatively, press `Delete` to delete the test run results. For tagged builds, you have to verify that you want to remove the results.
	View Manual Test Results	Click to view the **Current Run** page in read-only mode.
Execution Plan	The name of the execution plan that was executed. This column is only displayed for configuration suites.	
Status	Status summary of the run. A bar lists the amount of passed, failed, and not executed tests. The run status of each assigned test is shown in the second section of the page.	
Run ID	Identifier of the execution plan run. Click to access the results of the run.	
Keywords	Keywords assigned to the execution plan.	
Executed By	Name of the execution server on which the run was executed. For manual tests the name of the person who executed the run is listed.	
Errors	Number of errors that occurred in the run.	
Warnings	Number of warnings that occurred in the run.	
Product	The application under test.	
Version	The version is either set by the user for this execution plan, or is read from the build information file at execution time. This information can be set in **Administration** > **Products, Versions, and Builds** > **Products**.	
Build	The build is either set by the user for this execution plan, or is read from the build information file at execution time. This information can be set in **Administration** > **Products, Versions, and Builds** > **Products** . If the build is tagged, it is marked bold and with .	
Start Time	Time the run started.	
Duration	Time consumed to execute all included tests. Includes the duration of the setup and cleanup test and the time consumed to fetch automation files and code coverage, launch execution tools, and other tasks. For manual executions this is the time between starting and finishing the execution.	
Start Type	Shows how the test run was started. Manually, through a Web Service, or from a schedule.	
Starter Name	Name of the schedule, tester, or Web Service user.	

Column	Description
Start Scope	The scope specified in the **Run** dialog box.
Run Comment	You can use this column to add information to the run. To edit the comment, right-click on the run and select **Edit Run Comment**.

The test runs section lists the test runs for the selected execution plan run. The section is paged with fifty runs shown on a page. Use the arrow buttons to navigate through the pages. The page shows the following columns for each run:

Column	Description	
Actions	Actions that you can perform on the test run.	
	View or download results	If the test, to which the run belongs, is of a test type that generates result files, click on the icons to view or download the result files.
	New Issue	Click to open the **New Issue** dialog box and create a new issue for the test.
	Assign External Issue	Assign an issue from an externally-configured issue-tracking system to the test.
Run ID	Identifier of the test run. Click to open the **Test Run Results** dialog box.	
ID	Identifier of the test. This column is hidden by default.	
Test	Name of the test. Click to access the test in **Tests** > **Details View**. The icon corresponds to the test type.	
Start Time	Date and time the run started.	
Duration	Duration of the test run in `h/mm/ss`.	
Status	Status summary of the run. For a single test a single status is shown. A bar lists the amount of passed, failed, and not executed tests for a test package or suite node.	
Executed By	Name of the execution server on which the run was executed. For manual tests the name of the person who executed the run is listed.	
Issues Found	Displays the amount of issues that are assigned to the test run. When no issues are assigned to the test run, the column is empty. Click on the link to access the issue in the **Issues** page of **Tests** > **Details View**.	
Errors	Number of errors that occurred in the run.	
Warnings	Number of warnings that occurred in the run.	
Run Comment	For SilkPerformer test runs, SilkPerformer uses this column to add information to the test run when uploading results. For all other test types, you can use this column to add information to the run. To edit the comment, right-click on the run and select **Edit Run Comment**.	

Current Run Page

Execution Planning > **Details View** > **<Execution Plan>** > **Current Run**

Note: The **Current Run** page displays the active manual test run, until the test run is finished. For manual tests that are assigned to a testing cycle, this page is renamed to **Run** page. In such a case, the **Run** page continues to display the run, after the run is finished.

To access the **Current Run** page of an execution plan, select the execution plan in the **Execution Plans** tree and click the **Current Run** tab.

The **Current Run** page shows information about the active manual test run and enables you to see who is working on which test step. You can execute the manual test run with a few clicks. The page features two grid views, **Assigned Tests** and **Test Steps**. **Assigned Tests** shows information about the active manual test run, and **Test Steps** shows information for each of the steps in the manual test. You can filter the test runs in the **Assigned Tests** view by selected columns, through right-clicking the desired column. When multiple runs of a manual test are in progress, the one first started is shown. Three additional views display additional information on the execution plan, the active test, and the selected test step.

The two grid views offer view settings that are configurable on a per-user basis. Display or hide columns, adjust the width of columns, and move columns around by clicking on a column and dragging it to the desired location. Use **Ctrl+Click** or **Shift+Click** to select multiple tests or test steps using standard browser multi-select functions. Right click on your selection and set the status of the selected test steps to the selected status.

You can perform the following general actions on the **Current Run** page:

Action	Description
Reload	Click to reload the **Current Run** page.
Synchronize Run	Click to update the tasks shown in the **Current Run** page. The following items are updated: **Item** — **Description** **Assigned tests** — Only when you start the run with scope **Run all Tests** and click **Synchronize Run**, tests that are newly assigned to the execution plan are shown, and tests that are no longer assigned to the execution plan are removed, if the tests are not already started. **Test properties** — The name, the description, the attachments, and the other test properties of the assigned tests. **Test step properties** — The name, the description, the attachments, and the other test-step properties of the steps in the assigned tests.
Select Build	Click to change the version or build number for a running execution plan.
Finish Run	Click to finish the test execution and to open the **Finish Run** dialog box. From the **Build** list box, choose the appropriate build. The build of the execution plan on which the manual test was started is preselected from the list. If there exist tests in the execution plan run with status `Not Executed`, you can choose an action to perform from the **Status** list box. The following items are available in the dialog box: **Item** — **Description** **Remove tests from this run** — Removes all tests with status `Not Executed` from the execution plan run. This action is only available if the run includes tests with status `Not Executed`.

Action	Description	
	Item	**Description**
	Set to Passed, Failed, Unsupported, or Unresolved	Sets all tests with status `Not Executed` to the selected status. This action is only available if the run includes tests with status `Not Executed`.
	Select Build	Select the build number for the execution plan run from the list box.

Note: You can show or hide the **Execution Plan Run Details** view by clicking on the arrows in the top-right corner. The view is collapsed by default.

Execution Plan Run Details

This area displays the following information on the execution plan run, which has the active manual test run assigned:

- The ID of the execution plan run.
- The name of the execution plan.
- The configuration suite or folder in the context of which the execution plan is executed. Click to access the suite or folder in the **Execution Plans** tree. If the execution plan is not included in a configuration suite or folder, nothing is displayed.
- The keywords assigned to the execution plan run.
- The version is either set by the user for this execution plan, or is read from the build information file at execution time.
- The build is either set by the user for this execution plan, or is read from the build information file at execution time.
- The run type of the execution plan run.
- The start type of the execution plan run.
- The name of the tester, schedule, or Web Service user that started the execution plan run.
- The start scope of the execution plan run, specified in the **Run** dialog box.
- The start time of the execution plan run.

Assigned Tests

This area provides the following information for the manual test run:

Columns	Description	
Actions	Actions that can be performed during the run. Some of these actions are not available for data-driven tests. The following actions can be performed:	
	Add Result File	Add a result file to the test.
	New Issue	Create a new issue for the test.
	Assign External Issue	Assign an issue from an externally-configured issue-tracking system to the test.
	Edit Test	Edit the test in the **Edit Test** dialog box. When you close the dialog box, Test Manager automatically synchronizes the test run.

Columns	Description
#	The order of the test in the execution plan run.
ID	The identifier of the test. This column is hidden by default.
Test	The name of the test. Click on the name to view the test, or to perform an action on the test.
Status	The current status of the test. Click on the status to change it.
Executed By	The user that last worked on this test execution. For example entering step results, adding attachments, and so on.
Used Time [hh:mm]	The actual execution time. Click the field to note how long it actually took to execute the test. Type in the time in the `hh:mm` format. If you type in just a number, it will be considered to be hours. When the run or testing cycle is finished, the entered time will be displayed in the **Duration** column.

Test Steps

This area provides the following information for each test step:

Actions	Actions that can be performed during the run.	
	Edit	Edit the test step in the **Edit Step** dialog box. When you close the dialog box, Test Manager automatically synchronizes the test run.
	New Issue	Click to display the **New Issue** dialog box allowing you to associate an issue with the current step.
	Assign External Issue	Click to open the **Assign External Issue** dialog box which allowing you to associate an external issue with the current step.
#	Order of the step in the test.	
Step Name	Name of the step. Click to access the step in **Tests** > **Details View** > **Steps** .	
Status	Execution status of the step. Click on the status to change it.	
Result	Result of the step. Click on the text box to edit the result.	

Tip: You can hide the **Test Steps** view by clicking on the arrows in the top-right corner.

Test Details

This area displays the following information for the active manual test:

- The name of the test.
- The planned time for the execution of the test (manual tests only).
- The amount of attachments attached to the test. Click on the link to access the attachments.
- The amount of issues assigned to the test. Click on the link to access the issues.
- Result files generated for the test. Click on the links to access the result files.

  Note: Click the **Delete** icon to delete a result file.
- The description of the test.

Step Details

This area displays the following information for the selected step:

- The name of the step.
- A description of the action the step performs.
- The expected result of the step.
- The result of the step, when the test run is finished.
- The attachments to the step.

Tip: You can hide the **Step Details** view by clicking on the arrows in the top-right corner.

If other execution plan runs are started while the **Current Run** page is open, a note displays, stating that newer runs are available. You can see information on those runs in the **Activities** page.

For automated tests, the **Current Run** page shows the progress of the execution.

Working with SilkPerformer Projects

SilkPerformer is fully integrated with the test and execution functionality of Test Manager. SilkPerformer projects can be integrated into Test Manager tests and directly executed through Test Manager. This allows for powerful test-result analysis and reporting. It also enables unattended testing, which means tests are run automatically by Test Manager based on pre-configured schedules.

Refer to the SilkPerformer Help for details on configuring the integration of SilkPerformer with Test Manager.

SilkPerformer project files can be directly opened in SilkPerformer from Test Manager, where scripts and settings can be edited. Edited SilkPerformer projects can subsequently be checked back into Test Manager to make them available for future test executions.

Test Manager provides information on execution plan run properties during SilkPerformer test executions. Use the `AttributeGet` methods to access execution plan run properties in the SilkPerformer script. You can access the following properties in the script:

- `#sctm_execdef_id`
- `#sctm_execdef_name`
- `#sctm_product`
- `#sctm_version`
- `#sctm_build`
- `#sctm_keywords`

Note: The term Project is used differently in SilkPerformer than it is in Test Manager. A SilkPerformer project, when uploaded to Test Manager, becomes the core element of a Test Manager test. Test Manager projects are high-level entities that may include multiple SilkPerformer projects, tests, execution plans, and requirements.

Downloading SilkPerformer Projects

Whereas opening a SilkPerformer project may involve checking out a SilkPerformer project from a source-control tool, editing the project in SilkPerformer, and checking the project back into Test Manager, *downloading* a project involves downloading a copy of a project and working with it independently of Test Manager. Changes you make to downloaded projects are not automatically migrated back to Test Manager.

To download a SilkPerformer project:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.

4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click . A file download dialog box displays, asking you to confirm that you wish to download the specified SilkPerformer project to your local system.
6. Click **Save** to open the file in SilkPerformer. If not already open in the background, SilkPerformer is invoked.
7. The **Select Target Directory** dialog box displays, loaded with the default directory path to which the specified SilkPerformer project will be saved. If you approve of the specified pathname, click **OK**, otherwise click **Browse** to specify an alternate path.

Note: Even if you have configured source-control integration, you will not be prompted to check out the SilkPerformer project from your source-control system because you are working with this file independently of Test Manager.

Note: SilkPerformer projects utilized by Test Manager can also be downloaded directly from the SilkPerformer user interface. For additional information, refer to the SilkPerformer documentation.

Opening SilkPerformer Projects

To open a SilkPerformer project from Test Manager:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.
4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click . A file download dialog box displays, asking you to confirm that you wish to open the specified SilkPerformer command file (.spwbcmd) in SilkPerformer.
6. Click **Open** to open the file in SilkPerformer. If not already open in the background, SilkPerformer is invoked. The **Select Target Directory** dialog box opens, loaded with the default directory path to which the specified SilkPerformer project will be saved.
7. If you approve of the specifed pathname, click **OK**, otherwise click **Browse** to specify an alternate path.
8. If you have configured source-control integration for Test Manager, for example Visual SourceSafe, you are presented with a login screen for your source-control client. Enter valid user connection settings and click **OK** to continue.

Note: SilkPerformer projects utilized by Test Manager can also be opened directly from SilkPerformer. For additional information, refer to the SilkPerformer documentation.

Executing Attended SilkPerformer Tests

Attended tests are SilkPerformer tests that are executed manually in SilkPerformer and are not executed automatically based on a pre-defined schedule in Test Manager.

Note: To use Test Manager's data-driven test functionality with SilkPerformer scripts, data sources with column names matching the corresponding SilkPerformer project, attributes must be used in conjunction with `AttributeGet` methods.

To execute an attended test run in SilkPerformer:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.
4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click . A file download dialog box displays, asking you to confirm that you wish to run the specified SilkPerformer command file (.spwbcmd).
6. Click **Open** to open the project in SilkPerformer. If not already open in the background, SilkPerformer is invoked. The **Select Target Directory** dialog box displays, loaded with the default directory path to which the specified SilkPerformer project will be saved.

7. If you approve of the specified pathname, click **OK**, otherwise click **Browse** to specify an alternate path. The **SilkPerformer Workload Configuration** dialog box opens with all of the workload settings that are associated with the SilkPerformer project.
8. Edit the workload settings as required and click **Run** to begin the test and monitor the test results with SilkPerformer.

   Note: Clicking **Run** without editing any workload settings executes the SilkPerformer test in exactly the same way as if the test had been executed directly from Test Manager as an unattended test.

Editing SilkPerformer Test Properties

Note: To use the data-driven test functionality of Test Manager with SilkPerformer scripts, you have to use data sources with column names that match the corresponding SilkPerformer project attributes in conjunction with `AttributeGet` methods.

To edit SilkPerformer test properties:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.
4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click **Edit SilkPerformer Test Properties**.
6. Proceed with the configuration of your SilkPerformer test.

Analyzing SilkPerformer Test Results

Performance Explorer enables in-depth analysis of SilkPerformer test results. The **Analyze Results** option downloads only selected results, in contrast to “Downloading result packages”. To assist you in analyzing the results of your optimization efforts, Performance Explorer even allows you to compare statistics from multiple test runs side-by-side in cross load-test reports.

The results of tests that are run using Test Manager can be automatically loaded into Performance Explorer through commands on the **Runs** page in the **Tests** area.

For full details on using Performance Explorer and integrating Performance Manager with Test Manager, refer to the Performance Explorer documentation.

To open SilkPerformer test results in Performance Explorer:

1. In the menu, click **Tests** > **Details View** .
2. Select the test you are interested in viewing.
3. Click the **Runs** tab.
4. Click in the **Actions** column of the test execution for which you want to download results. A **File Download** dialog box displays, showing you the name of the Performance Explorer command file, `.sppecmd`, that you are about to download.
5. Click **Open** to open the results in Performance Explorer. Alternatively, you can click **Save** to save the results locally.
6. If not already open in the background, Performance Explorer now opens, connected directly to your Test Manager installation, and fetches the results of the selected execution run.

   Note: To prepare for a cross load-test report that compares the results of multiple executions in a single report, you may download the results of additional executions from the **Runs** page. Additional execution results are displayed in the existing instance of Performance Explorer on the Performance Explorer Test Manager tab. For additional details on cross load-test reports, refer to the Performance Explorer documentation.

Downloading SilkPerformer Test Result Packages

Downloading result packages is the ideal option if you want to analyze the complete results set of a test run, or if you want to download the complete results set for offline analysis. Because result packages often include large TrueLog On Error files, result packages can be compressed and downloaded to your local hard drive as `.lrz` files. Downloading results locally can also be useful when you are working from a slow Internet connection.

To download SilkPerformer test results:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Runs** tab.
4. Click in the **Actions** column of the test execution for which you want to download results. A **File Download** dialog box displays, showing you the name of the compressed results package file, `.ltz` that you are about to download.
5. Click **Open** to open the results in Performance Explorer. Alternatively, you can click **Save** to save the results locally.
6. If not already open in the background, Performance Explorer now opens. You are presented with an **Import Project** dialog box that indicates the target directory to which the results will be saved. Click **OK** to accept the default path, or click **Browse** to select an alternate path. The downloaded results are then displayed in Performance Explorer.

   Note: If you accept the default projects directory where result packages are typically stored, which we generally recommend, then the results will be stored with all otherSilkPerformer results and will be readily accessible through the Performance Explorer `Add Loadtest Results` command.

Uploading SilkPerformer Test Results

Once you have completed running an attended test in SilkPerformer, you can upload the test results to Test Manager and associate the results with a test.

To upload results from an attended SilkPerformer test:

1. Run an attended SilkPerformer test.

   For additional information, see *Executing Attended SilkPerformer Tests*.
2. When the test is complete, select **Upload Results to Test Manager** from the **Results** menu.

   The **Login** screen of the **Upload Results to Test Manager** wizard displays.
3. Enter your **Password** and click **Next**.

   Note: Because this is an attended test, the wizard already knows the appropriate hostname and username of the test to which these results are to be uploaded.
4. If not already selected by default in the project list, select the Test Manager project to which you want to upload the SilkPerformer results.
5. If not already selected by default in the tree list, select the test to which you want to upload the results. Click **Next**.

   Note: You can right-click in the tree and use the commands on the context menu to create a new test, child test, test folder, or child test folder to which the results can be saved.
6. On the subsequent screen you can specify Version and Build numbers for the assigned Product to which the uploaded results belong. Also specify the SilkPerformer test result status, for example `Passed`, `Failed`.

   Note: If any errors occurred during the test run, the test result status is set to `Failed` by default.
7. Click **Finish** to upload the results. The uploaded results are displayed in the **Test Runs** column in **Tests** > **Details View** > **<SilkPerformer Test>** > **Runs**.

SilkTest Classic Tests

This section describes how to execute tests in SilkTest Classic.

Automated Execution of SilkTest Classic Tests

All tests within an execution plan use the same SilkTest Classic instance for tests. The SilkTest Classic GUI is opened once with the first SilkTest Classic test execution and is closed automatically after the last SilkTest Classic test execution. Each SilkTest Classic test execution produces its own results. If for any reason the SilkTest Classic GUI closes during a test, it will reopen automatically with the next SilkTest Classic test execution.

Automated Execution of Data-Driven SilkTest Classic Testcases

If the **data driven** check box is checked in SilkTest Classic test properties, each SilkTest Classic test will be repeated once for each data row in the external datasource. By default, plan-based execution mode is used for data driven tests. This means that the results of all data rows will be listed under a single node in the `.res` result file. If execution mode is switched to script-based data driven in `SccExecServerBootConf.xml`, a result node will be created in the `.res` result file for each data row.

Specifying Agent Under Test (AUT)

When a SilkTest Classic agent cannot run on the same machine as the Test Manager execution server, for example when tests are run on platforms other than Windows, the hostname and port may be specified by the SilkTest Classic AUT Hostname setting in the **Deployment** page of an execution plan. If the setting has not been defined, SilkTest Classic default values are used, for example from `partner.ini`. The syntax for AUT is `hostname:port`. The agent must be started manually prior to test execution and configured to listen at the specified port. By default, the TCP/IP protocol is used for communication between SilkTest Classic instances and SilkTest Classic agents. Ensure that both programs have been configured to use the same protocol.

Note: Be careful when you have multiple execution servers assigned to an execution plan as SilkTest Classic agents can only work with one SilkTest Classic instance at a time.

SilkTest Classic Time-out Settings

If you have SilkTest Classic test cases that require more than 1 hour to complete, you must adjust the time-out settings in Test Manager. Otherwise, Test Manager assumes that something has gone wrong in the execution and terminates SilkTest Classic. For details about setting the SilkTest Classic time-out, refer to the *SilkCentral Administration Module Help*.

SilkTest Classic Logs

The RMS log file in SilkTest Classic is used to log data for each test case as test runs progress. Three types of data records are written to this file: status, memory and user records. By monitoring this file, the RMS Remote Agent has a means of determining the progress of each test run.

You can write your own comments into the user records of the log file by executing the `PrintToRMSLog` 4Test function.

Examples:

```
PrintToRMSLog ("Error", "An intended error")
PrintToRMSLog ("Info", "testcase sleep1 started")
PrintToRMSLog ("Warning", "TestCase 1 started a second time")
```

Definition of user function in `rms.inc`:

```
PrintToRMSLog (STRING sMessageType, STRING sUserMessage)
```

Writes to the log file in the following format:

```
U|{sTestCaseName}|{sScriptName}|{sArgStr}|{sUserMessage}|{sMessageType}
```

VMware Lab Manager Integration

This section explains the integration of Test Manager with VMware Lab Manager (Lab Manager).

VMware Lab Manager Virtual Configurations

VMware images are virtual computer systems. Lab Manager is used to manage VMware images, or "configurations", which are combinations of images, for example database server, application server, and execution server. VMware configurations offer an effective means of virtualizing complex software-testing lab environments. Configurations are typically deployed from Lab Manager libraries, and are turned on and off just like individual VMware images. Multiple instances of the same configuration can be deployed simultaneously, with separate tests run in each instance. VMware configurations are “network-fenced,” meaning that they do not influence each others' network behavior. VMware LiveLink technology enables you to take “snapshots” of complete configurations that can later be recreated (or “restored”) on demand.

The integration of Lab Manager with Test Manager enables users to manage Lab Manager directly from the Test Manager UI. Integrated functionality includes configuration deployment, test execution, result collection, and automatic undeployment of configurations. Test Manager can support multiple Lab Manager installations and configurations. Configurations captured using LiveLink technology are viewed using VMware Lab Manager. Refer to the VMware Lab Manager documentation for full details regarding LiveLink configuration captures and other Lab Manager functionality. For details on configuring the integration of Test Manager with Lab Manager, refer to the *SilkCentral Administration Module Help*.

Note: At least one Test Manager execution server must exist within each configuration. These execution server instances control test execution within configurations and retrieve test results.

Note: Lab Manager users must have administrator rights to access the Lab Manager API.

Tracking

The **Tracking** area provides project-level information including **Activities**, **Cross-Project Activities**, **Project Overview Report**, and **Quality Goals**.

Activities

This section explains how to manage upcoming, current, and recently-executed test runs.

Deleting Last Executions Runs

To delete a run from the **Last Executions** list:

1. In the menu, click **Tracking** > **Activities**.
2. In the **Last Executions** area of the **Activities** page, right-click the test run you want to delete and select **Delete Run Results**.
3. Confirm the deletion by clicking **OK**.

Displaying or Hiding Columns on the Activities Page

To display or hide columns on the **Activities** page:

1. In the menu, click **Tracking** > **Activities**.
2. Right-click a column header.
3. Expand the **Columns** submenus to view all the columns that are available in the project.

4. Select the check boxes of all the columns that you want to have displayed. Your column-display preferences will be saved and displayed each time you open the active project.

Entering Issues From the Activities Page

New issues can be entered directly on the **Activities** page.

To enter an issue from the **Activities** page:

1. In the menu, click **Tracking** > **Activities**.
2. In the **Last Executions** area, click the **Run ID** of the relevant execution plan to view test-execution results.

   Each test associated with the execution run is listed in the **Assigned Tests** table at the lower part of the view.
3. In the **Actions** column of the test to which you want to associate the issue, click **Create a new issue for this test**.
4. Proceed with defining the issue.

Filtering Test Runs on the Activities Page

You can filter the views on the **Activities** page based on column values. You can specify filter strings to be applied to:

- Text-based data fields.
- Calendar filters (using **Before**, **After** or **On** operators) for date-based fields.
- Numerical operators (**>**, **<**, and **=**) for number-based fields.

Filtering Text-Based Values on the Activities Page

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the text-based column that the filter is to be based on.
3. Expand the **Filter** submenu on the context menu to display the **Filter** text box.
4. Enter a text string into the text box.
5. Press **Enter**. All entries that match the filter criteria are dynamically displayed in the filtered list.

Filtering Date-Based Values on the Activities Page

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the date-based column that the filter is to be based on.
3. Hold your cursor over **Filter** on the context menu to display the **Before**, **After**, or **On** submenu.
4. Choose from the following:
   a) Hold your cursor over **After** to define a date before which (and including) all entries should be excluded.
   b) Hold your cursor over **Before** to define a date after which (and including) all entries should be excluded.
   c) Hold your cursor over **On** to exclude all entries except those that have the specified date.

   The calendar tool displays.
5. Select a date using the calendar tool (or click **Today** to specify today's date).

   Tip: You must explicitly click a date on the calendar tool or press **Enter** to activate date-based filtering changes.

   All entries that match the filter criteria are dynamically displayed in the filtered list.

Filtering Number-Based Values on the Activities Page

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the number-based column that the filter is to be based on.
3. Expand the **Filter** submenu on the context menu to display the **> (greater than)**, **< (less than)**, and **= (equals)** operators.
4. Choose from the following:
   a) Enter a number in the **>** text box to define a number less than which (and including) all entries should be excluded.
   b) Enter a number in the **<** text box to define a number greater than which (and including) all entries should be excluded.
   c) Enter a number in the **=** text box to exclude all entries except those that have the specified number.

   Note: Number values are rounded to two decimal places.
5. Press **Enter**. All entries that match the filter criteria are dynamically displayed in the filtered list.

Filtering Boolean Values on the Activities Page

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the boolean-based column that the filter is to be based on.
3. Expand the **Filter** submenu on the context menu to display the available values.
4. Click one of the **Yes** or **No** option buttons. All entries that match the filter criteria are dynamically displayed in the filtered list.

Filtering Values Using a Predefined List on the Activities page

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the column that has a predefined filter value that the filter is to be based on.
3. Expand the **Filter** submenu on the context menu to display the **Filter** text box.
4. Check the check boxes of the filter values that you want to use. All entries that have one of the selected criteria will be displayed.

Grouping Test Runs on the Activities Page

Beyond simply sorting by column, you can chunk entries into groups to facilitate viewing. Groups are based on commonly-shared values within the column that grouping is based on.

Applying Groups to Test Runs on the Activities Page

To group entries on the **Activities** page:

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the column that the sort is to be based on.
3. Select **Group by This Field**.

   Entries are then organized into groups based on commonly-shared values within the column you have selected.

Removing Grouping from Test Runs on the Activities Page

To remove grouping:

1. In the menu, click **Tracking** > **Activities**.
2. Right-click any column.
3. Uncheck the **Show in Groups** check box.

Removing Activities Filters

Note: Hiding a column removes all filters that have been applied to the column.

To remove one or more filters:

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the column that has the filter you want to remove.

   Note: You can identify filtered columns by their titles, which are displayed in bold, italic text.
3. Do one of the following:
   a) To remove a specific filter: Uncheck the **Filter** check box.
   b) To remove all filters: Select **Reset Filters**.

Reordering Columns on the Activities Page

To reorder columns on the **Activities** page:

1. In the menu, click **Tracking** > **Activities**.
2. Select the column header of the column you want to move.
3. Drag the column to the desired position and release it.

   Your column-order preferences will be saved and displayed each time you open the active project.

Resizing Columns on the Activities Page

To adjust the width of columns on the **Activities** page:

1. In the menu, click **Tracking** > **Activities**.
2. Select the vertical column-header divider of the column you want to adjust.
3. Drag the column boundary to the desired position and release it.

   Your column-width preferences will be saved and displayed each time you open the active project.

Restoring Default Activities Page View Settings

Restoring default view settings resets all user-defined settings, which are column order, column width, shown/hidden columns, applied filters, sorting, and grouping, for the current project.

To restore default view settings:

1. In the menu, click **Tracking** > **Activities**.
2. Right-click any column header.
3. Select **Reset View**.

Sorting Test Runs on the Activities Page

To sort test runs on the **Activities** page:

1. In the menu, click **Tracking** > **Activities**.
2. Right-click the header of the column you want the test runs to be sorted by.
3. Select **Sort Ascending** to have the test runs sorted in ascending order or select **Sort Descending** to have the test runs sorted in descending order. Your sort preferences will be saved and displayed each time you open the active project.

Activities Page

Tracking > **Activities**

The **Activities** page offers a centralized location from which you can manage upcoming, current, and recently executed test runs for a project. The grid views on the **Activities** page offer filtering, sorting, and grouping options that are configurable for each user separately. You can display or hide columns, adjust the width of columns, and move columns around using click and drag.

The **Activities** page is split into three sections: **Next Executions**, **Current Executions**, and **Last Executions**. The grid views can be resized by dragging the separators between the views.

Context-sensitive menu commands are available for each test run. These commands enable you to link directly to listed execution plans, continue manual tests, manage test-run results, and more.

The **Activities** page makes it easier to identify match points between execution plans and to find specific execution plan information. Standard Windows keyboard shortcuts can be used to select test run entries, making it easy to select and manipulate specific sets of execution plans and test results. Sorting, grouping, and filtering functions are available through context-menu commands to help you better organize and group test runs. All of your view-customization preferences are saved along with your project and will be available to you each time you visit the **Activities** page.

Note: Data on the **Activities** page is not automatically refreshed. Click **Reload** near the paging buttons at the lower part of each view to refresh the page's contents.

Note: You can use `CTRL + SHIFT` to select multiple queued executions and abort them all with one click.

Next Executions

The **Next Executions** view lists the execution plans that are scheduled to run in the future. To enhance performance when you have numerous execution plans, only the upcoming 50 execution plans are displayed. You can access additional future execution plans by using the available filtering features. To edit an execution plan, right-click the execution plan and choose **Go to Execution Plan** or click on the arrow to the left of the name of the execution plan. This takes you to the **Executions** area where you can view and edit the details of the execution plan.

By default, all execution plans are sorted by **Start Time**. Columns in the **Next Executions** view can not be sorted or grouped.

Next Executions view can be collapsed and expanded by clicking the double-arrow button on the right-hand side of the view's title bar.

For each execution plan, the **Next Executions** view displays the following columns:

Column	Description
ID	Identifier of the scheduled execution plan. This column is hidden by default.
Execution Plan/Folder	Name of the scheduled execution plan or folder.
Keywords	Keywords that are assigned to the scheduled execution plan.
Manual Testers	For manual tests, this column contains the user name of the manual tester that is assigned to the test. This column is blank when no manual testers are assigned to the test.
Priority	The priority that has been assigned to the execution plan.
Start Time	Scheduled start time of the test run.

Column	Description
Execution Plan Parent	The configuration suite or folder in the context of which the execution plan is executed. Click to access the suite or folder in the **Execution Plans** tree. If the execution plan is not included in a configuration suite or folder, nothing is displayed.

Current Executions

The **Current Executions** view lists the execution plans that are currently running (both automated and manual test runs).

To abort an execution plan that is currently in progress, click **Abort** in the **Actions** column of the execution plan. To view or edit an execution plan, right-click the execution plan and select **Go to Execution Plan** or click on the arrow to the left of the name of the execution plan. To view the execution's progress, right-click the automated execution plan and select **View Details** or click the execution plan's **Run ID/Task ID** link.

As long as a manual test remains open, the corresponding execution plan remains in the list of **Current Executions** with a status of **Pending**. Right-click a manual execution plan and select **Continue Manual Test**, or click the execution plan's **Continue Manual Test** button in the **Actions** column, to enter manual step results and statuses in the **Manual Test Execution** view.

To view the results of a manual test, right-click the manual execution plan and select **View Details**, or click the execution plan's **Run ID/Task ID** link, to open the **Results for Execution Plan** page. From there, click the name of the manual test in the **Assigned Tests** section to open the **Results** dialog box. Detailed results of the manual test are displayed there. Back on the **Results for Execution Plan** page, click **Manual Test Results** to go to **Manual Test** view , where information about the status of the assigned manual test is available.

Page views of current executions are broken into views of 20 execution plans each. You can advance through pages using **First**, **Last**, **Next**, and **Previous** located in the lower part of the **Current Executions** view. Or you can type a page number into the **Page** text box and press `Enter`.

For each execution plan, the **Current Executions** view displays the following columns:

Column	Description	
Actions	You can perform the following actions on the execution plan:	
	Abort	Click to cancel the current execution. Alternatively, press `Delete`. When you abort executions, these executions are grayed out until the background process completes the deletion. **Note:** If the execution is part of a testing cycle the abort action is disabled. The icon and the entry in the context menu are grayed out and such runs are ignored when you abort multiple runs.
	Continue Manual Test	Click to go to **Manual Test** view and to enter manual step results and statuses.

Column	Description
	Manual Testing Client Click to open the Manual Testing Client. This button is available to users that are assigned as testers to the selected test.
ID	Identifier of the execution plan. This column is hidden by default.
Execution Plan	Name of the execution plan.
Run ID/Task ID	Manual tests receive a **Run ID** when they are executed. When the manual test is completed, the **Run ID** carries over to the **Last Executions** view. Automated tests receive a **Task ID** when they are executed. The **Task ID** is not carried over to the **Last Executions** view. Completed execution plans receive a **Run ID** in the **Last Executions** view.
Status	Status of the active execution plan or manual test. For automated tests, the status is indicated with a text-based value, `Pending` or `Active`. For manual tests, the status is indicated with a colored histogram. Automated-test statuses are described textually and can be filtered. Manual tests can be filtered by checking relevant properties on the **Filter** submenu.
Keywords	Keywords that are assigned to the execution plan.
Executed By	• For manual tests, this column displays the users who are assigned to perform the manual test. This column is blank when no manual testers are assigned to the test. • For automated executions, this column displays the name of the execution server.
Priority	For automated execution plans, when more than one execution plan is queued but only one execution server is available, the **Priority** determines which execution plan is executed first.
Start Time	Time when the execution plan run started.
Time Left	Amount of time remaining until the test is complete. For manual tests that do not have an estimated time, this column has a value of `unknown`. This time is not available for executions on Lab Manager virtual machines, and if the test is executed the first time since the used execution server was started.
Start Type	Shows how the test run was started. • Manually • Through a Web Service • From a schedule • From a dependency
Starter Name	Name of the schedule, tester, or Web Service user.

Column	Description
Start Scope	The scope defined in the **Run** dialog box.
Execution Plan Parent	The configuration suite or folder in the context of which the execution plan is executed. Click to access the suite or folder in the **Execution Plans** tree. If the execution plan is not included in a configuration suite or folder, nothing is displayed.

Last Executions

The **Last Executions** view lists all past execution plan runs, except deleted runs, for which results were collected from the execution server. You can filter the listed execution plan runs, for example by the start time.

To view or edit an execution plan, right-click the execution plan run and choose **Go to Execution Plan**, or click on the arrow to the left of the name of the execution plan. To display the **Results for Execution Plan** page of an execution plan run, right-click the run and select **View Details**, or click the **Run ID** link of the execution plan. This page shows details for the selected execution plan run and includes any files and messages, for example LiveLink VMware configuration captures, that were generated during the execution. To open the **Results** dialog box of a test, click on the **Run ID** of the test in the **Assigned Tests** portion of the **Results for Execution Plan** page.

To compare two execution plan runs, use `Ctrl + Shift` to select the two runs. Right click on your selection and click **Reports** > **Execution Plan Run Comparison ...** .

Right-click on an execution plan and click **Run Execution Plan** to run the execution plan. Test Manager uses the configuration currently defined in the **Executions** area for the re-run, not the original configuration of the selected execution plan run.

For execution plans that are deployed to virtual servers: To open VMware Lab Manager and restore a captured LiveLink configuration, expand the **Messages** link on an execution plan run's **Results for Execution Plan** page and select **LiveLink**.

To delete an execution plan run, right-click a run entry and choose **Delete Results** or click **Delete** in the **Actions** column of the run.

Test-result page views are broken into views of 20 test results each. You can advance through pages using **First**, **Last**, **Next**, and **Previous** at the bottom of the **Last Executions** view. Or you can enter a page number into the **Page** text box and press `Enter`.

The **Last Executions** view can be collapsed/expanded by clicking the double-arrow button on the right-hand side of the view's title bar.

For each execution plan, the **Last Executions** view displays the following columns:

Column	Description	
Actions	You can perform the following actions on the execution plan:	
	Delete	Click to delete the execution plan run results. When you delete executions, these executions are grayed out until the background process completes the deletion. Alternatively, press `Delete` to delete the execution plan run. For tagged builds, you have to verify that you want to remove the results.

Column	Description
	View Manual Test Results — Click to open the **Manual Test** view page; or right-click a manual execution plan run and select **View Manual Test Results**.
ID	The identifier of the executed execution plan. Unassigned tests have an ID value of N/A. This column is hidden by default.
Execution Plan	Name of the executed execution plan. Click on the name to view or edit the execution plan.
Run ID	ID assigned to the test run. Click the link to view details of the test run.
Status	Displays the number of passed, failed and not executed tests in a bar graph. Filtering, sorting, and grouping are not available for the column.
Keywords	Keywords that were assigned to the execution plan at execution time.
Executed By	• For manual tests, this column displays the users who are assigned to perform the manual test. This column is blank when no manual testers are assigned to the test. • For automated executions, this column displays the name of the execution server.
Start Time	Time when the execution plan run started.
Duration	Time consumed to execute all included tests. Includes the duration of the setup and cleanup test and the time consumed to fetch automation files and code coverage, launch execution tools, and other tasks. For manual executions this is the time between starting and finishing the execution.
Product	The product under test. This column is hidden by default.
Version	The version that either is set by the user for this execution plan, or is read from the build information file at execution time. This column is hidden by default.
Build	The build that either is set by the user for this execution plan, or is read from the build information file at execution time. If the build is tagged, it is marked bold and with .
Start Type	Shows how the test run was started. Manually, through a Web Service, or from a schedule.
Starter Name	Name of the schedule, tester, or Web Service user.
Start Scope	The scope specified in the **Run** dialog box.
Execution Plan Parent	The configuration suite or folder in the context of which the execution plan is executed. Click to access the suite or folder in the **Execution Plans** tree. If the execution plan is not included in a configuration suite or folder, nothing is displayed.

Cross-Project Activities Page

Tracking > **Cross-Project Activities**

The **Cross-Project Activities** page enables a user with SuperUser privileges to see all execution related activities across projects. It provides all the options the **Activities** page offers, and additional options across projects. This allows some conclusion about the execution queue. The SuperUser can remove executions from the queue to resolve bottlenecks.

The **Cross-Project Activities** page is visible to the SuperUser only. The page is split into the same three sections as the **Activities** page: **Next Executions**, **Current Executions**, and **Last Executions**. All three sections include an additional column with the **Project ID**.

Item	Description
Project ID	The ID of the project to which the execution plan belongs to.

Project Overview Report

Test Manager > **Tracking** > **Project Overview Report**

The *Project Overview Report* contains a high-level overview of the status of the selected project.

Quality Goals

Quality goals (exit criteria) are a definable set of testing metrics that the project must meet in order for the quality to be considered acceptable. Each Test Manager project can contain one or more quality goals. Test Manager uses quality goals as a primary means to implement risk based testing, allowing you to plan, to test, and to report on only the most important tests in a given testing cycle.

In Test Manager, a quality goal is comprised of the following two elements:

- A requirement or test.
- One of the following:
  - A custom attribute or property of type **List** with a corresponding percentage value.
  - One of the following default requirement properties: **Priority**, **Reviewed**, or **Risk** with a corresponding percentage value.

Quality Goal Example

If a requirement has a property of `MyCustomRisk` with values of `High`, `Medium`, and `Low`, it should be possible to define a quality goal of:

Requirement Goal - MyCustomRisk	`High = 100%`
	`Medium = 50%`
	`Low = 10%`

The `100%` means that all of the tests that meet that criteria (linked to a requirement that has the *High* attribute value) must execute and pass to meet that goal.

Permissions

Quality goals are permission based. See *Quality Goals Permissions* for details of permission.

Quality Goals Page

Tracking > **Quality Goals**

The following controls are available on the **Quality Goals** tab allowing you to create, edit, or delete quality goals for a project:

Quality Goals Grid

Column	Description
Goal Name	The name of the goal. Select the drop-down list on the column name and then select **Show in Groups** to show all goals in a list.
Name	Each value of the attribute or property `List` type is itemized in the **Name** column.
Goal (%)	This text field must contain a numeric value from `0` to `100`. The value represents the percentage of tests required to meet the defined quality goal.

Buttons

- **Manage Goals** - Click to show the **Manage Quality Goals** dialog box.
- **Planning Report** - Click to show the **Quality Goals Planning Report**.
- **Execution Report** - Click to show the **Quality Goals Execution Report**.

Adding Quality Goals

Before creating a quality goal, you need create a test custom attribute or requirement custom property of type `List` that you want to use for your quality goals. Or you can use the requirement fields of **Priority**, **Reviewed**, or **Risk**.

For example, if you want a quality goal to track a requirement custom property with values of `High`, `Medium`, and `Low`, first go to **Projects:<Project Name>** > **Project Settings** > **Requirement Properties** and create a `List` type custom property that contains these values.

Note: You can only use a property or attribute as a quality goal once per project.

1. In the menu, click **Projects** > **Project List**.
2. Select a project.
3. Click **Tracking** > **Quality Goals**.
4. Click **Manage Goals**. The **Manage Quality Goals** dialog box opens.
5. In the **New Goal Name** field, enter the name for your goal.
6. In the **Attributes/Custom Properties** list, select the item that contains the values that you want to use for your quality goal. The list contains the following types of items:
   - Requirement property of type **List**.
   - Default requirement properties of **Priority**, **Reviewed**, or **Risk**.
   - Test attribute of type **List**.
7. Click **Add New Goal**.
8. Click **Close**. The **Manage Quality Goals** dialog box closes and the new goal is added to the **Quality Goals** grid.
9. For each value in the list, enter a numeric value from `0` to `100` in the **Percentage of Tests Required to Meet Goal**.

Deleting Quality Goals

To delete a quality goal:

1. In the menu, click **Projects** > **Project List**.
2. Select a project.

3. Click **Tracking** > **Quality Goals**.
4. Click **Manage Goals**. The **Manage Quality Goals** dialog box opens.
5. Click the **Delete** icon next to the quality goal that you want to delete.

Analyzing Results

SilkCentral Test Manager enables you to analyze the results of your tests.

Video Capturing

You can enable video capturing for a test element or an execution plan. The videos are added as a WMV video file to the result files in the **Files** page of the **Test Run Results** dialog box. You can then analyze the videos, for example if errors have occurred during tests.

The **Video Recording** setting in the **Deployment** page of an execution plan can override the setting for a specific test element, which is included in the execution plan.

Note: Video capturing is currently enabled only for execution servers with Microsoft Windows as their operating system. To use video capturing on a Windows 2008 execution server, you need to install the following server features:

- Desktop Experience
- Quality Windows Audio Video Experience

Attention: The process of capturing videos is very CPU-intensive. Use this functionality only when needed. To use this functionality you require a system with sufficient CPU power, for example a system with multiple CPUs.

Capturing a video for each individual test case in a suite is supported out of the box for JUnit 4 tests and SilkTest Classic Plan tests. To use this feature for other test types, use the methods in the `TestCaseStartFinishSocketClient` interface. For additional information, see *Indicating Start and Finish for Video Capturing*.

Enabling Video Capturing for a Test

To enable video capturing for a test:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a container, folder, or test.
3. Click the **Parameters** tab.
4. Click **Add Custom Parameter**. The **Set Custom Parameter** dialog box opens.
5. In the **Name** text box, type `recordVideo`.
6. From the **Type** list box, select `String`.
7. In the **Value** text box, type one of the following:
   - `always`, if you want to add a video to the result files on every execution of the execution plan.
   - `onError`, if you want to add a video to the result files when an error occurs during the execution of the execution plan.
   - Any other string or nothing, if you do not want to add a video to the result files.

Screen Capturing

You can enable screen capturing for a test element or an execution plan. The screenshots are added to the result files in the **Files** page and the **Timeline** page of the **Test Run Results** dialog box. You can then analyze the screenshots, for example if errors have occurred during tests.

The **Screen Capturing** setting in the **Deployment** page of an execution plan can override the setting for a specific test element, which is included in the execution plan.

Enabling Screen Capturing for a Test

To enable screen capturing for a test:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a container, folder, or test.
3. Click the **Parameters** tab.
4. Click **Add Custom Parameter**. The **Set Custom Parameter** dialog box opens.
5. In the **Name** text box, type `captureScreen`.
6. From the **Type** list box, select `String`.
7. In the **Value** text box, type one of the following:
   - `onError`, if you want to add a screenshot to the result files when an error occurs during the execution of the execution plan.
   - Any other string or nothing, if you do not want to add a screenshot to the result files.

Issues

Describes how to work with issues in Test Manager.

Creating New Issues

The **Issues** page enables you to easily enter issues related to the selected test.

To create a new issue:

1. In the menu, click **Tests** > **Details View** .
2. Select the test for which you want to create a new issue.
3. Click the **Issues** tab.
4. Click **New Issue** to open the **New Issue** dialog box.
5. Select the **Profile** of the issue-tracking system you are submitting the issue to.

   The **Profile** list box shows the always available internal issue-tracking profiles and any external issue-tracking profiles that you may have created. Select **Internal** to save the issue only to the Test Manager database. Select an external profile to have the new issue saved to both the external tool and Test Manager.

   Note: The profile you select here becomes the default selection for when you enter new issues in the future. When adding a new issue to an external tracking system, you will be prompted to provide login credentials for the external system. The credentials that you provide will be automatically preselected for you in the future. If you do not provide credentials, the default credentials stored in the profile will be used.
6. Enter a brief **Synopsis** of the issue.
7. Enter a meaningful **Description** of the issue.
8. Specify the **Status** of the issue.
   - `Open`
   - `Fixed`
   - `Verified`
   - `Deferred`
   - `Closed`

   When using an external profile, status is set by the external tool.

9. Specify the **ExternID** of the corresponding issue in the external issue-tracking profile.

   Note: The ExternID is the corresponding issue ID in the external tool. This option is disabled when you have specified an external issue-tracking profile because the external tool sets this value. When the Internal profile is selected, this value can be set manually.

10. Specify the **ExternLink** of the issue-tracking profile.

    Note: The ExternLink is the HTTP link to the issue in the external tool. This option is disabled when you have specified an external issue-tracking profile because the external tool sets this value, when the tool offers direct HTTP links to issues, as is the case with Issue Manager. When a link is specified, the ExternID is shown as a link in the issue list. Depending on the issue-tracking profile you are working with, the **New Issue** dialog box may include other tracking fields that are specific to the external issue-tracking tool.

11. Click **Save** to save the new issue.

    Note: Issue Manager determines the ID numbers of newly created issues.

Viewing Issue Statistics in Document View

To view issue statistics in **Document View**:

1. In the menu, click **Issues** > **Document View**.
2. In the **Issues** tree, select the project, issue-tracking system, or product for which you want to view statistics.

Issues Document View

Issues > **Document View**

The **Document View** displays issue statistics for the selected project in tabular format.

Column	Description
Date	Date and time when issue details were updated.
Open	Number of issues in the selected project, database, or product that have a status of `Open`.
Fixed	Number of issues in the selected project, database, or product that have a status of `Fixed`.
Verified	Number of issues in the selected project, database, or product that have a status of `Verified`.
Closed	Number of issues in the selected project, database, or product that have a status of `Closed`.
Deferred	Number of issues in the selected project, database, or product that have a status of `Deferred`.

Viewing Issue Statistics in Details View

To view issue statistics in **Details View**:

1. In the menu, click **Issues** > **Details View**.
2. In the **Issues** tree, select the project, issue-tracking system, or product for which you want to view statistics.
3. The calendar tool feature enables you to specify the time period over which you want to view issue statistics. Click the time-frame dates link to expand the calendar.

For additional information, see *Specifying a Calendar Range*.

4. Using the calendar's **From** and **To** list boxes, specify start and end times for issue statistics.
5. Click **Update** to update the chart view based on the specified time range.

Issues Page

Issues > **Details View** > **Issues**

The **Issues** page lists issues from both internal and external databases that are configured for the selected project. The page includes the following items:

Item	Description
Calendar Tool	Allows you to specify a time-frame for which issues should be reported.Click the time-frame date link to expand the calendar tool. For additional information, see *Specifying a Calendar Range*.
Update	Updates the **Issues View** based on calendar changes. The button is displayed only when changes exist.

Assigning External Issues

The **Issues** page enables you to assign issues from externally configured issue tracking systems to a selected test.

To assign an existing external issue to a test:

1. In the menu, click **Tests** > **Details View** .
2. Select the node of the test for which you want to assign an external issue.
3. Click the **Issues** tab.
4. Click **Assign External Issue** to open the **Assign External Issue** dialog box.
5. Select the profile of the pre-configured, external issue-tracking system where the issue is tracked.
6. In the **External ID** text box, type the unique alpha-numeric ID of an existing issue in the external issue-tracking system.
7. Click **OK**.

Note: You can also assign an existing external issue to a currently running execution plan in the **Current Run** page, the **Execution Runs** page, and the **Test Runs** page.

Synchronizing Internal and External Issue States

To synchronize issue states between Test Manager and an external issue tracking-system:

1. In the menu, click **Tests** > **Details View** .
2. Select the node of the test for which you are updating a corresponding external issue.
3. Click the **Issues** tab.
4. Click **Update states of external Issues** to synchronize the state of the issues listed in Test Manager with the corresponding issues in the external tool.

Deleting Issues

To delete an issue from the issue-tracking system:

1. In the menu, click **Tests** > **Details View** .
2. Select the test from which you want to delete an issue.

3. Click the **Issues** tab.
4. In the Actions column of the issue that you want to delete, click ✕.
5. Click **Yes** on the **Delete Issue** dialog box to confirm the deletion.
6. External issues are not effected when internal issues are deleted.

Specifying a Calendar Range

To view issues within a specific calendar time frame:

1. In the menu, click **Issues** > **Details View**.
2. Select an issue in the menu tree.
3. Click ▷ in the top-left corner of the tab view to open the calendar.
4. Specify the **From** and **To** date/time for which you want to view issues by clicking the respective ✎.
5. Click **Update** to refresh the page with the issue listings that are included within the specified time frame.

Calendar Tool

The calendar tool provides the following features:

◁, ▷	These buttons move the time frame forward or backward in time at an interval roughly equivalent to the current timeframe. For example, if the current timeframe encompasses about 1 week, clicking ▷ will advance the timeframe into the future one week.
🔍−	Increases the timeframe by 50 percent so that more test executions are included in the list.
🔍+	Decreases the timeframe by 50 percent so that fewer test executions are included in the list.
◁ day ▷	Moves the selected timeframe backward or forward one day.
◁ week ▷	Moves the selected timeframe backward or forward one week.
◁ month ▷	Moves the selected timeframe backward or forward one month.
◁ quarter ▷	Moves the selected timeframe backward or forward one quarter.
Last 7 days	Sets the past 7 days as the selected timeframe.
Last 31 days	Sets the past 31 days as the selected timeframe.

Integrations

SCTM integrates with many different tools for many different purposes. The following topics detail integration details for all SCTM integrations.

Code Analysis Tools Integration

DevPartner Code Coverage Integration

Test Manager integrates with the **Code Analysis** component of DevPartner Studio. You can perform code analysis on .NET applications using this integration.

For more information, see the sections on code analysis for .NET.

Issue Tracking Profile Integrations

Atlassian JIRA

This section describes how to configure Atlassian JIRA (JIRA) issue tracking profiles to integrate with Test Manager.

The JIRA plug-in relies on the SilkCentral Java API for integration. Test Manager communicates with JIRA through the JIRA Web-service by using the Apache Axis library.

Note: See the sources of the package `com.borland.sctm.issuetracking.jira` to see how these elements fit together.

For a list of the JIRA versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding Atlassian JIRA Issue Tracking Profiles

To integrate Test Manager with JIRA, enable the System Soap Services plug-in in JIRA. For detailed information, refer to the JIRA documentation.

To add a JIRA issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Atlassian JIRA** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. In the **Endpoint** field, type the hostname of your JIRA server and the port that is used to connect to the server.
9. Click **Load Project** to load all projects from the server and populate the **Project** list box, then select a project from the **Project** list box.
10. Optional: Select **true** from the **Show custom fields** list to display custom JIRA fields in the issue dialog. If true is selected, the JIRA user defined for the issue tracking profile has to have JIRA administration permissions.
11. Optional: If you experience performance issues on large JIRA installations, select **false** from the **Update Daily Issue Statistics** list. The daily issue statistics will not be updated.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

13. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing Atlassian JIRA Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing Atlassian JIRA issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. In the **Endpoint** field, edit the hostname of your JIRA server and the port that is used to connect to the server.
9. To change the JIRA project, click **Load Project** to load all projects from the server and update the **Project** list box, then select a project from the **Project** list box.
10. Optional: Select **true** from the **Show custom fields** list to display custom JIRA fields in the issue dialog. If true is selected, the JIRA user defined for the issue tracking profile has to have JIRA administration permissions.
11. Optional: If you experience performance issues on large JIRA installations, select **false** from the **Update Daily Issue Statistics** list. The daily issue statistics will not be updated.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
13. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Bugzilla

This section describes how to configure Bugzilla issue tracking profiles to integrate with Test Manager.

The Bugzilla plug-in relies on the SilkCentral Java API for integration. Test Manager communicates with Bugzilla through the XML-RPC Bugzilla Web-service introduced with Bugzilla 3.0 by using the Redstone

XML-RPC library. To enable Bugzilla integration, ensure that you have installed a SOAP::Lite package for the web server. For example, for the Debian distribution install the `libsoap-lite-perl` package.

For a list of the Bugzilla versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding Bugzilla Issue Tracking Profiles

To add a Bugzilla issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Bugzilla** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Enter the URL of your Bugzilla installation. For example, `http://bugzillaserver/cgi-bin/bugzilla3`.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
10. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing Bugzilla Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing Bugzilla issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.

7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the URL of your Bugzilla installation.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
10. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Changepoint

This section describes how to configure Changepoint issue tracking profiles.

Changepoint is an IT management and governance tool that enables organizations to maximize the business value of the entire IT portfolio including projects, applications, and infrastructure.

This integration allows you to connect Test Manager to Changepoint to store issues.

Important: The Changepoint environment must have the Changepoint API installed and the `CPWebService` configured.

Known Issues

Workflow	Do not change the Changepoint workflows for requests after submitting them in Test Manager. Do not use different workflows in requests when assigning external issues. The filter criteria used in workflows should be based on fields used in the Test Manager **Issue Tracking Profile**. Otherwise, unexpected states may be assigned to requests in Test Manager.

Creating a Changepoint Issue Tracking Profile

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Compuware Changepoint** from the **Type** list box.
7. Type a valid **Username** and **Password** for the issue tracking system.

   Note: In order to select a project and to submit a new issue, the **Username** field must contain a Changepoint user who has the project's `Edit Project Plan` permission.
8. Enter the URL of your Changepoint installation in the **Changepoint Server URL** field.
9. Enter the URL of your Changepoint web service server in the **Changepoint WebService URL** field.
10. Click **Load Initiator** to load your Changepoint initiators.
11. Select an initiator from the **Initiator** list box.

12. Click **Load Client** to load your Changepoint clients.
13. Select a client from the **Client** list box.
14. Click **Load Initiative** to load your Changepoint initiatives.
15. Select an initiative from the **Initiative** list box.
16. Click **Load Project** to load your Changepoint projects.
17. Select a project from the **Project** list box.
18. Click **Load Request Type** to load your Changepoint request types.
19. Select a request type from the **Request Type** list box.
20. Click **Load Application** to load your Changepoint applications. Select an application from the **Application** list box.
21. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
22. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Opening Changepoint Issues in Context

In order to open Test Manager issues in the Changepoint system, you need to follow the steps below to install a connection file on the Changepoint server.

1. In Test Manager, click **SilkCentral** > **Help** > **Tools** to view the **Downloadable Client Tools** page.
2. Click the `Changepoint Request Form` link.
3. When prompted, choose to save the `ChangepointRequestForm.zip` file to disk.
4. Extract `CPRequestForm.html` from `ChangepointRequestForm.zip`.

   If you are using Changepoint 2010, the required file is in the `Changepoint 2010` folder. It is called `CPRequestForm2010.html`. Extract this file and manually rename it to `CPRequestForm.html`.
5. Manually copy `CPRequestForm.html` to the Changepoint server and placed into the virtual directory in which Changepoint is running.

Synchronizing the Time for the Changepoint and Test Manager Servers

This section describes how to ensure that the Changepoint server and Test Manager server communicate properly.

1. Open the `web.config` file located in the `../Changepoint/CP Web Services` directory.
2. Add the following key to the `security` section:

   ```
   <timeToleranceInSeconds>86400</timeToleranceInSeconds>
   ```
3. Save the file.

IBM Rational ClearQuest

This section describes how to configure IBM Rational ClearQuest issue tracking profiles to integrate with Test Manager.

IBM Rational ClearQuest products provide flexible defect/change tracking and automated workflow support. The two key products are IBM Rational ClearQuest (ClearQuest) and IBM Rational ClearQuest MultiSite (ClearQuest Multisite). To work with ClearQuest profiles, you must have the ClearQuest client

software installed on the computer where the SilkCentral front-end server is running. For detailed information about installing ClearQuest, refer to the ClearQuest documentation.

For a list of the ClearQuest versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding ClearQuest Issue Tracking Profiles

To add a ClearQuest issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **IBM Rational ClearQuest** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Enter the **Repository Info** of your ClearQuest installation.

   This is the database name that is defined in the ClearQuest client software.
9. Specify the **Record Type**, which is the issue type of ClearQuest.

   When entering an issue in Test Manager, ClearQuest will save the issue with the issue type you define in this text box. The default issue type is `Defect`.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing ClearQuest Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing ClearQuest issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

This is the name that is displayed in the Test Manager GUI.

5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the **Repository Info** of your ClearQuest installation.

   This is the database name that is defined in the ClearQuest client software.
9. Change the **Record Type**, which is the issue type of ClearQuest.

   When entering an issue in Test Manager, ClearQuest saves the issue with the issue type you define in this field.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

SilkCentral Issue Manager

SilkCentral Issue Manager (Issue Manager), the issue-tracking tool of SilkCentral, is fully integrated with Test Manager, enabling you to correlate issues with system requirements and executed tests.

Test issues can be added and managed in the menu through **Tests** > **Details View** > **Issues**. For more information, refer to the Issue Manager documentation.

Note: StarTeam and IBM Rational ClearQuest are also supported by Test Manager “out of the box”. Additional issue tracking systems can be configured by installing a custom plug-in. For more information, refer to the *Test Manager API Help*.

Adding SilkCentral Issue Manager Issue Tracking Profiles

To add a SilkCentral Issue Manager issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Issue Manager** from the **Type** list box, or select **Issue Manager 3.3** to connect to an Issue Manager version 3.3 installation.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Type the **Issue Manager URL** of your Issue Manager installation. This is the URL you use to login to Issue Manager, though without the `login` extension at the end.

For example, if your standard Issue Manager URL is `http://IssueManager/login`, then the correct service URL is `http://IssueManager`.

9. If you selected **Issue Manager 3.3** from the **Type** list box, proceed with the next step.

   If you selected **Issue Manager** from the **Type** list box, proceed as follows:

   1. Click **Load Projects**. This action will populate the **Project** list box with all initialized Issue Manager projects to which the specified user has access to. Note that only those projects display for which Issue Manager user groups have been defined, and the defined user is a member of at least one user group.
   2. Select the **Project** where Issue Manager issues are maintained.

   Caution: We recommend not to use identical projects for Issue Manager and Test Manager, as this limits flexibility in working with both tools on different future projects.

10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing SilkCentral Issue Manager Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing Issue Manager issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.

5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.

8. Edit the **Issue Manager URL** of your Issue Manager installation if the location has changed.

   This is the URL you use to login to Issue Manager, though without the `login` extension at the end. For example, if your standard Issue Manager URL is `http://IssueManager/login`, then the correct service URL would be `http://IssueManager`.

9. To change the StarTeam project, click **Load Project** to load all projects from the server and update the **Project** list box, then select a project from the **Project** list box.
10. Click **OK**.

Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

11. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

StarTeam Issue Tracking Profiles

This section describes how to configure StarTeam issue tracking profiles to integrate with Test Manager.

StarTeam is a software change management and configuration management tool that enables coordination and management of the software delivery process.

To work with StarTeam profiles and use the go-to-link functionality for change requests in StarTeam, you must have the StarTeam Cross-Platform Client software installed on the computer where the browser is running.

For a list of the StarTeam versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding StarTeam Issue Tracking Profiles

To add a StarTeam issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Type a **Description** for the new profile.
6. Select **StarTeam** from the **Type** list box
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Type the **Hostname** of your StarTeam server and the **Port** that is used to connect to the server. If this setting has not been changed, use the default port `49201`.
9. Specify the type of **Encryption** that the profile supports.
10. Click **Load Project** to load all projects from the server and populate the **Project** list box, then select a project from the **Project** list box.
11. Click **Load View** to load all views for the selected project and populate the **View** list box, then select a view from the **View** list box.
12. Click **Load Status Field** to load all enumeration fields for change requests and populate the **Status Field** list box, then select a status field from the **Status Field** list box.

    If you are using a custom workflow in StarTeam, this field is the workflow driver field in StarTeam that maps to the Test Manager issue state.
13. Select a type of link from the **Link Type** list box.

starteam:// **External ID** links on the **Issues** tab will open the cross platform client.

http:// **External ID** links on the **Issues** tab will open the issue in the StarTeam web UI.

14. If you selected `http://` in the **Link Type** list box, enter the web server address of the StarTeam web UI in the **WebServer** field.
15. Select `Yes` or `No` in the **Workflow** field. Select `Yes` to show the required fields that are specified in the workflow of the selected **View**.
16. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
17. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing StarTeam Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing StarTeam issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the **Hostname** of your StarTeam server and the **Port** that is used to connect to the server.
9. Modify the type of **Encryption** that the profile supports.
10. To change the StarTeam project, click **Load Project** to load all projects from the server and update the **Project** list box, then select a project from the **Project** list box.
11. To change the view, click **Load View** to load all views for the selected project and populate the **View** list box, then select a view from the **View** list box.
12. To change the workflow driver field, click **Load Status Field** to load all enumeration fields for change requests and populate the **Status Field** list box, then select a status field from the **Status Field** list box.

    If you are using a custom workflow in StarTeam, this field is the workflow driver field in StarTeam that maps to the Test Manager issue state.
13. Select a type of link from the **Link Type** list box.

    starteam:// **External ID** links on the **Issues** tab will open the cross platform client.

http:// **External ID** links on the **Issues** tab will open the issue in the StarTeam web UI.

14. If you selected `http://` in the **Link Type** list box, enter the web server address of the StarTeam web UI in the **WebServer** field.
15. Select `Yes` or `No` in the **Workflow** field. Select `Yes` to show the required fields that are specified in the workflow of the selected **View**.
16. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
17. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Team Foundation Server

This section describes how to configure Team Foundation Server (TFS) issue tracking profiles to integrate with Test Manager.

For a list of the TFS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Installing a Team Foundation Server Web Service Proxy

To communicate with a TFS, you need to install a Team Foundation Server Web Service Proxy as an interface to the TFS.

To install a TFS proxy service:

1. Install Microsoft Visual Studio Team Explorer 2010 from *http://www.microsoft.com/downloads/details.aspx?FamilyID=fe4f9904-0480-4c9d-a264-02fedd78ab38*.
2. Install IIS and the Web Deployment Tool, for example with the Web Platform Installer from *http://www.microsoft.com/web/downloads/default.aspx*.
3. Download the Team Foundation Server Web Service Proxy from **Help** > **Tools**.
4. Unzip the downloaded package.
5. Open a command shell and type `DotNetTfsWebServiceProxy.deploy.cmd /Y` to install the proxy service. If an error message box displays, stating that ASP.NET 4 is required, refer to *http://msdn.microsoft.com/en-us/library/k6h9cz8h.aspx*.
6. Open the IIS Manager.
7. Verify that the new website exists.
8. In the root folder of the virtual directory, open the `Web.config` file.
9. In the *appSettings* section of the file, modify the value of the key *WorkItemTrackingCacheRoot* to a local directory.

   For example:

```
<appSettings>
  <add key="WorkItemTrackingCacheRoot" value="C:\temp" />
</appSettings>
```

10. If the directory you have specified does not exist, create it.
11. Grant **Everyone** full rights on the specified directory.

For additional information on deploying an ASP.NET Web Application, refer to *http://msdn.microsoft.com/en-us/library/ee942158.aspx*.

Adding TFS Issue Tracking Profiles

You need to configure a Team Foundation Server Web Service Proxy, which is provided by Micro Focus to enable Test Manager to access TFS issues.

Important: This is not the Team Foundation Server Proxy provided by Microsoft.

To add a TFS issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **Team Foundation Server 2010** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Type the **Domain** of the TFS user.
9. Enter the URL of your TFS installation.

   For example, `http://tfsserver:8080/tfs`.
10. Type the **Collection** to which your project belongs.

    For example `DefaultCollection`.

    All projects in the collection are listed in the **Projects** list box.
11. Select the **Project** from the list box.
12. Type the URL of your TFS proxy. For example `http://tfsproxyserver/DotNetTfsWebServiceProxy_deploy/TfsWebServiceProxy.asmx`.
13. Click **Load Work Item Type**. The **Work Item Type** list box is populated with the available work item types.
14. Select the **Work Item Type** from the list box.
15. Click **Load Initial State**. The **Initial State** list box is populated with the states that are allowed for the selected work item type.
16. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
17. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.

    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Editing TFS Issue Tracking Profiles

When the server or login credentials of your issue tracking system change, you must edit your issue tracking profile accordingly.

To edit an existing TFS issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click on the name of the issue tracking profile that you want to modify. The **Edit Issue Tracking Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **Description** of the profile.
6. *Optional:* Select a new **Type** for the issue tracking profile from the list box.
7. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.
8. Edit the **Domain** of the TFS user.
9. Edit the **URL** and **Collection** of your TFS installation.
10. Select a different **Project**.
11. Edit the URL and port of your TFS proxy.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

Other Integrations

Integrating Test Manager into Rally

To use the project management tool Rally with Test Manager, you have to integrate Test Manager into Rally.

A mashup is shipped with Test Manager, that enables you to display user-story related test-coverage information from Test Manager in Rally. You can link each test in Test Manager to one or more user stories in Rally. To link tests to a user story, assign an attribute with the value of the formatted user-story ID in Rally to each test.

Creating a Mashup Tab in Rally

Create an HTML or JavaScript mashup tab in Rally to display user-story related test-coverage information.

To create a mashup tab:

1. Navigate to the tab in Rally where your mashup tab should reside.
2. Click **New Custom Tab**.
3. Type a name for your new tab in the **Name** text box. For example, `Iteration Status`.
4. *Optional:* Check the **Share with all Users** check box if you wish other users to be able to see your mashup.
5. Select **Custom Mashup** as the type of the mashup.
6. In the menu, click **Help** > **Tools** .
7. Download the **Rally Iteration Status Mashup**.
8. Paste the content of the **Rally Iteration Status Mashup** into the **HTML** text box.
9. Configure the script in the text box to match your Test Manager environment:

- Check if the valid script source is set, for example `<script type="text/javascript" src="http://localhost:19120/silkroot/script/sctm-toolkit.js"></script>`.
- Check if the valid URL for Test Manager is set, for example `var SCTM_URL = "http://localhost:19120/"`.
- Check if a valid Test Manager project ID is set, for example `var SCTM_PROJECT_ID = 0`.
- Check if a valid attribute name is set, for example `var SCTM_ATTRIBUTE_FOR_RALLY_US = "rallyattr"`.
- Make sure usage of the `FormattedID` from Rally is enabled, `var USE_FORMATTED_ID = true`.

10. Click **Save & Close**. The mashup table is created.

Linking User Stories from Rally to Tests

Link Rally user stories to tests in Test Manager.

To link Rally user stories to Test Manager tests:

1. Select the project in Test Manager which contains the tests you want to link the Rally user stories to.
2. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
3. Click the **Attributes** tab.
4. Click **New Attribute**. The **New Attribute** dialog box opens.
5. Type `rallyattr` in the **Name** text box.
6. Select **Edit** from the **Attribute type** radio box.
7. Click **OK** to create the new attribute.
8. Assign the new attribute to each test that you want to link user stories from Rally to.
9. Set the value of the attribute to be the value of the **FormattedID** of the appropriate Rally user story for each test. If you assign multiple stories to a single test, separate the IDs with a comma and put no whitespace characters between them. For example, `US10,US12,US14`.

VMware Lab Manager Integration

This section explains the integration of Test Manager with VMware Lab Manager (Lab Manager).

VMware Lab Manager Virtual Configurations

VMware images are virtual computer systems. Lab Manager is used to manage VMware images, or "configurations", which are combinations of images, for example database server, application server, and execution server. VMware configurations offer an effective means of virtualizing complex software-testing lab environments. Configurations are typically deployed from Lab Manager libraries, and are turned on and off just like individual VMware images. Multiple instances of the same configuration can be deployed simultaneously, with separate tests run in each instance. VMware configurations are “network-fenced,” meaning that they do not influence each others' network behavior. VMware LiveLink technology enables you to take “snapshots” of complete configurations that can later be recreated (or “restored”) on demand.

The integration of Lab Manager with Test Manager enables users to manage Lab Manager directly from the Test Manager UI. Integrated functionality includes configuration deployment, test execution, result collection, and automatic undeployment of configurations. Test Manager can support multiple Lab Manager installations and configurations. Configurations captured using LiveLink technology are viewed using VMware Lab Manager. Refer to the VMware Lab Manager documentation for full details regarding LiveLink configuration captures and other Lab Manager functionality. For details on configuring the integration of Test Manager with Lab Manager, refer to the *SilkCentral Administration Module Help*.

Note: At least one Test Manager execution server must exist within each configuration. These execution server instances control test execution within configurations and retrieve test results.

Note: Lab Manager users must have administrator rights to access the Lab Manager API.

SAP Solution Manager Integration

Describes how to configure and to use the integration between Test Manager and SAP Solution Manager application management solution (SAP Solution Manager).

SAP Solution Manager facilitates technical support for distributed systems, covering solution deployment, operation, and continuous improvement. SAP Solution Manager is a centralized, robust application management and administration solution, that combines tools, content, and direct access to SAP to increase the reliability of solutions and lower total cost of ownership. For additional information on the SAP Solution Manager, refer to the SAP Solution Manager documentation.

The integration between Test Manager and SAP Solution Manager allows for the synchronization of SAP Solution Manager business processes and incidents as Test Manager requirements and issues.

System Requirements for SAP Solution Manager Integration

The following are the system requirements for the SAP Solution Manager integration plugin:

- SilkCentral Test Manager 2011 with hotfix 1 or later.
- SAP Solution Manager 7.01 Patch 24 or later.
- The *SCTM Integration Hub for SAP Solution Manager* must be installed in SAP Solution Manager.

Configuring SAP Solution Manager

Configure SAP Solution Manager to enable the integration with Test Manager.

For additional information, see `Configuration_Guide_SilkCentral_Integration_Hub_for_SAP.docx` and `User_Guide_SilkCentral_Integration_Hub_for_SAP.docx`, which are included in the SAP Solution Manager plug-in.

For this integration to get valid links from SAP Solution Manager to Test Manager, the front-end server machine of Test Manager must have a valid fully qualified name.

Adding SAP Solution Manager Issue Tracking Profiles

To add a SAP Solution Manager issue tracking profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Issue Tracking** tab. The **Issue Tracking** page opens, listing all of the issue tracking profiles that have been created for the system.
3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Type a **Description** for the new profile.
6. Select **SAP Solution Manager** from the **Type** list box.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access the issue tracking system.
8. Type the **URL** of the `Z-SCTM_ADAPTER` web service of SAP Solution Manager.
9. Click **Load Project** to load all projects from the server and populate the **Project** list box, then select a project from the **Project** list box.
10. Specify the **System** with which to integrate.

11. Click **Load Default Incident Type** to load all possible incident types and populate the **Default Incident Type Field** list box.
12. Select an incident type from the **Default Incident Type Field** list box. If you are creating links to existing incidents in SAP Solution Manager and you do not specify the incident type, this type will be used for it.
13. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.
14. If the trial connection is successful, a confirmation dialog box displays, asking you if you want to map internal issue states to the states of the newly defined profile.
    - Click **Yes** to proceed with the related *Mapping Issue States* procedure.
    - Click **No** to map issue states later.

Enabling Requirements Integration with SAP Solution Manager

1. In the menu, click **Projects** > **Project List**.
2. Select the project to which you want to establish integration.
3. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
4. Click the **Requirements Management** tab.
5. In the SAP Solution Manager section, click **Configure**.
6. Enter the **URL** of the `Z_SCTM_ADAPTER` web service of SAP Solution Manager.
7. Enter the **Username** and **Password**.
8. Click **Test Connection** to confirm that the host and user credentials are correct. If the settings are correct, a **Test connection was successful** message displays.
9. Click **Load** to load all available projects and select the correct one.
10. Optional: Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in SilkCentral Test Manager projects that are configured for integration.
11. Click **OK** to confirm and close the dialog.

Requirement Management Tools

IBM Rational RequisitePro

The topics in this section describe the integration between IBM Rational RequisitePro and Test Manager.

Enabling Integration with IBM Rational RequisitePro

To enable integration with IBM Rational RequisitePro:

1. In the menu, click **Projects** > **Project List**.
2. Select the project to which you want to establish integration.
3. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
4. Click the **Requirements Management** tab.

5. In the **IBM® Rational® RequisitePro® Integration** section, click **Configure**. The **Edit Configuration** dialog box opens.
6. Type the **UNC project path** to the machine where the external server is installed. Or click **Browse** and select the path.
7. Type valid **UNC Username** and **UNC Password** credentials for the machine where the external server is installed.
8. Type valid **Username** and **Password** credentials for the requirements management server.
9. Click**Test Connection** to confirm that the host and user credentials are correct. If the settings are correct, a **Test connection was successful** message displays.
10. Click **OK**.

    Note: Consult your system administrator if you are not able to establish a connection.

11. Click **Edit Packages and Requirement Types**.

    Your selections are then displayed on the **Edit Configuration** dialog.

    Note: Only requirements of explicitly selected packages will be synchronized. Selecting a parent package does not select the child packages of the parent.

    The **Browse Packages & Requirement Types** dialog box opens. The packages and requirement types that are available with the selected project are automatically populated into the **Packages** and **Requirement Types** list boxes.
12. From the **Packages** list box, select one or more packages from the external project that should be integrated with the Test Manager project.

    Use `CTRL + CLICK` to select multiple packages.
13. From the **Requirement types** list box, select one or more requirement types from the external project that should be integrated with the Test Manager project.

    Use `CTRL + CLICK` to select multiple requirement types.
14. Click **OK**.The **Browse Packages & Requirement Types** dialog box closes.
15. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with RequisitePro.
16. *Optional:* Check the **Enable upload of requirements to RequisitePro** check box to enable the upload of unmapped or unassigned requirements from Test Manager to RequisitePro.This allows you to upload additional previously unmapped requirement trees to RequisitePro and then have those requirements mapped within Test Manager. The **Map Requirement** button in **Test Manager** > **Requirements** > **Properties** becomes enabled, allowing configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
17. Click **OK** to save your settings.

CaliberRDM Integration

The topics in this section describe the integration between CaliberRDM and Test Manager.

Note: If you change databases for a Test Manager instance that is integrated with CaliberRDM, you may not be able to perform a synch afterwards. If this happens, restart the front-end server and application server.

Note: If you are migrating from 2010 R1 to 2010 R2, your **Magnitude** (Text type) attribute used in tests created from synchronized requirements will be renamed to **Magnitude_Old**. For 2010 R2, **Magnitude** is a numeric attribute on which you can filter.

Enabling Integration with CaliberRDM

To enable integration with CaliberRDM:

1. From the Test Manager project to which you want to establish integration, click **Project:<Project Name>** > **Project Settings** > **Requirements Management** .
2. In the **CaliberRDM Integration** section, click **Configure**.

   The **Edit Configuration** dialog box opens.
3. Enter the **Hostname** of the machine where the external server is installed.
4. Enter the **Port** number on which the external server is listening.
5. Enter valid **Username** and **Password** credentials for the requirements management server.
6. Click **Test Connection** to confirm that the host and user credentials you have entered are correct.

   A **Connection successful** message box displays if the settings are correct. Click **OK** to continue.

   Note: Consult your system administrator if you are not able to establish a connection.
7. From the **Project name** text box, select the external project with which the Test Manager project is to be integrated.
8. Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with CaliberRDM.
9. Click **OK** to save your settings.

Generating Tests from External Requirements

From the RMSs currently shipped with Test Manager, only CaliberRDM supports generating tests. If you want to use another RMS to generate tests, the RMS must be able to generate tests. The Test Manager plug-in API includes a method to verify that the RMS is able to create tests. For more information, see *Requirements Plug-In API Interfaces*.

To generate tests from external requirements:

1. Establish integration with the external requirement system.

   For additional information, see *Enabling Integration with CaliberRDM*.
2. Select the requirement in the **Requirements** tree from which you wish to generate tests.

   Note: To be able to generate tests, the RMS plug-in needs to implement the `RMTestProvider` interface, and the interface method `isTestGenerationSupported` needs to return `true` for the type of the selected requirement.
3. Click **Generate Tests**. The **Generate Tests** dialog box displays.
4. Use **SHIFT + CLICK** or **CTRL + CLICK** to select the tests that you want to generate.
5. Click **Generate All** to generate all tests, or click **Generate Selected** to generate the selected tests. The **Tests** tree displays.
6. Select the folder to which you want to add the generated tests.

   Note: Tests that already exist in the selected destination folder are not created, but updated. No tests in the folder are deleted.
7. Click **OK**.

CaliberRM Integration

This section describes how to integrate CaliberRM with Test Manager.

Note: CaliberRM user defined attribute **Multiple selection user list** and **Multiple selection group list** must be mapped to the Test Manager text attribute type and not list type.

Enabling Integration with CaliberRM

To enable integration with CaliberRM:

1. In the menu, click **Projects** > **Project List**.

2. Select the project to which you want to establish integration.
3. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
4. Click the **Requirements Management** tab.
5. In the **Borland® CaliberRM Integration** section, click **Configure**. The **Edit Configuration** dialog box opens.
6. Type the **Hostname** of the machine where the external server is installed.
7. Type valid **Username** and **Password** credentials for the requirements management server.
8. Click **Test Connection** to confirm that the host and user credentials you have entered are correct.

   Click **Browse** to advance to the **Browse Projects** dialog box.

   If the settings are correct, a **Test connection was successful** dialog box opens.

   Note: If you are not able to establish a connection, consult your system administrator.
9. Click **OK**.
10. From the **Project** list box, select the external project with which the Test Manager project is to be integrated. The requirement types that are available with the selected project are automatically populated into the **Requirement Types** field. The baselines that are available with the selected project are automatically populated into the **Baseline** field.
11. Select a **Baseline** from the external project that should be integrated with the Test Manager project.

    Your selections are displayed on the **Edit Configuration** dialog box.
12. Click **OK**.
13. Select one or more requirement types from the external project that should be integrated with the Test Manager project.

    Use `CTRL + Click` to select multiple requirement types.
14. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with CaliberRM.
15. *Optional:* Check the **Enable upload of requirements to CaliberRM** check box to enable the upload of unmapped or unassigned requirements from Test Manager to CaliberRM. This allows you to upload additional previously unmapped requirement trees to CaliberRM and then have those requirements mapped within Test Manager. The **Map Requirement** button in **Test Manager** > **Requirements** > **Properties** becomes enabled, allowing configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
16. Click **OK** to save your settings.

Baseline Support for CaliberRM Integration

You can select the current baseline or existing user-defined baselines for CaliberRM integration. When you select a user-defined baseline, the **Map Requirement** button in **Requirements** > **Details View** > **<Requirement>** > **Properties** is disabled and requirements are not updated in the integrated CaliberRM project.

You cannot import modified baselined requirements into Test Manager. Requirements that are not of the current baseline can only be changed in CaliberRM if the version of the requirement that is used for the baseline is changed. Such changes are only updated within Test Manager requirements when a manual or scheduled synchronization is performed.

You can change a baseline after you import it into Test Manager. You can change the configured baseline to a different user-defined baseline or the current baseline. After such a change, the next synchronization of the baseline, either manual or scheduled, will update the Test Manager project and update, create, or delete requirements as required. When a baseline is changed, a message displays that states that the

changes will take effect after the next synchronization. When a baseline is changed from the current baseline to a user-defined baseline, a message displays informing you that, for user defined baselines, upload of requirements is disabled.

Note: With user-defined baselines, you can use the same requirement type of a CaliberRM project more then once. With current baselines, a requirement type of a CaliberRM project is allowed only once.

Handling Test Assignments in CaliberRM

Assigned tests are displayed, managed, and created as traces (“Trace to”) of synchronized requirements in CaliberRM.

When synchronized requirements are assigned to tests in Test Manager, the test assignments are pushed out to and displayed within CaliberRM. Conversely, when tests are assigned to synchronized requirements from CaliberRM, the assignments are pushed out to and displayed within Test Manager.

Copying CaliberRM-Integrated Projects

To manage CaliberRM baselines when copying Test Manager projects:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Requirements Management** tab.
3. Verify that the baseline you want to save is selected.

   Note: If a baseline is changed, you must perform a synchronization to update the project requirements with the baseline changes, before you can copy an associated Test Manager project. The integration configuration is only copied if you select a baseline other than the current baseline. If you select the current baseline, you need to specify if you want to keep the integration configuration in the original project or move it to the copied project.
4. If the baseline that you want to save is not selected, click **Edit Configuration**. The **Edit Configuration** dialog box displays.
5. Click **Browse** next to the **Project name** text box. The **Browse Projects** dialog box opens.
6. Select the baseline you want to save, then confirm your selection.
7. In the menu, click **Projects** > **Project List** . The **Projects** page displays, listing all existing projects and project baselines.
8. Click in the **Actions** column of the project you want to copy.

   Note: For full details on copying a project, refer to the *SilkCentral Administration Module Help*.

   The **Copy Project** dialog box displays.
9. Select the items you want to copy into the new project, then confirm your selection.
10. Apply the baseline that you want to continue working with to the Test Manager project.

    Note: After copying a project, the original project and the copy are identical. Define on which you will continue working on by applying the corresponding baseline.

Creating a Baseline for a CaliberRM-Integrated Project

To create a baseline of a CaliberRM-integrated Test Manager project:

1. Create a baseline of the Test Manager project.

   For more information on creating a baseline, refer to the *SilkCentral Administration Module Help*.

   A SilkCentral message box displays during the process, asking if you want to enable the CaliberRM integration for the new baseline.

2. Click **Yes**. When the new baseline is created, the **Baseline Project - Adapt Project Settings** dialog box opens.
3. Click **Edit** to change the requirements integration settings.

   The **Edit Configuration** dialog box opens.
4. Click **Browse** next to the **Project name** text box. The **Browse Projects** dialog box opens.
5. Select a CaliberRM project, a baseline, and the requirement type for synchronization.
6. Click **Ok** to close the **Browse Projects** dialog box.
7. Click **Ok** to close the **Edit Configuration** dialog box.
8. Click **Finish** to close the **Baseline Project - Adapt Project Settings** dialog box.

Telelogic DOORS Integration

This section describes how to integrate Test Manager and Telelogic DOORS® (DOORS).

Installing Telelogic DOORS on the Front-End Server

To integrate Test Manager and Telelogic DOORS, install the DOORS client on the Test Manager front-end server machine. If you use more than one front-end server machine, you must install the DOORS client to the same directory on each of the machines.

To install the DOORS client on the Test Manager front-end server machine:

1. In the menu, click **Help** > **Tools** .
2. Click **Test Manager Add-In for Telelogic DOORS** to download the DOORS plug-in package.

   The package contains two ZIP-Archives:

   - `DoorsRM.zip`
   - `DoorsClientLibs.zip`
3. Create a new folder with the name `testmanager` in the `…\lib\dxl` folder of your Telelogic DOORS client installation.

   The default path for this folder is `C:\Program Files\Telelogic\DOORS_8.20\lib\dxl\testmanager`.
4. Extract all DOORS script files from `DoorsClientLibs.zip` to this folder.

The plug-in package `DoorsRMPlugin.zip` is automatically installed to the `Plugins` folder of your Test Manager application server installation during the setup process. During startup of the application server, this plug-in will be published to all front-end servers.

Configure a Project for Integration with DOORS

To configure a project for requirements integration with DOORS:

1. From the Test Manager project to which you want to establish integration, click **Project:<Project Name>** > **Project Settings** > **Requirements Management** .
2. In the **Telelogic DOORS® Integration** section, click **Configure**.

   The **Edit Configuration** dialog box opens.
3. In the **RM service URL** text box, type the URL of the Test Manager DOORS requirement Web Service.

   The default value should point to the correct location already. For example `http://MySCTMHost:19120/services/doorsrequirementsmanagement`.
4. Type valid **Username** and **Password** credentials for the requirements management server.
5. The default DOORS client installation path is displayed in the **DOORS Installation Path** text box. If this path is not correct, click **Browse** to browse to and select the correct destination in the front-end server directory structure.
6. Click **Test Connection** to confirm that the host and user credentials you have entered are correct.

A **Connection successful** message box displays if the settings are correct. Click **OK** to continue.

Note: Consult your system administrator if you are not able to establish a connection.

7. Click **Browse** next to the **Project name** text box to open the **Browse Requirement Types** dialog box. From the **Project** text box, select the external project with which the Test Manager project is to be synchronized.

   The requirement types that are available with the selected project are automatically populated into the **Requirement types** text box. Select the requirement types that are to be synchronized and click **OK**. Use `CTRL + CLICK` to select multiple requirement types.
8. Your selections are now displayed on the **Edit Configuration** dialog box. Click **OK**.
9. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with Telelogic DOORS.
10. *Optional:* Check the **Enable upload of requirements to Telelogic DOORS®** check box to enable the upload of unmapped or unassigned requirements from Test Manager to DOORS.This allows you to upload additional previously unmapped requirement trees to DOORS and then have those requirements mapped within Test Manager. The **Map Requirement** button in **Test Manager** > **Requirements** > **Properties** becomes enabled, allowing configuration of top level requirements for external requirement types, which is required when uploading unmapped requirements.
11. Click **OK** to save your settings.

Caution: As the DOORS application object is used for communication, and this object does not support login data, but rather requires a running DOORS client, Test Manager starts each DOORS client process with the provided login data and then uses that same data for all subsequent application objects. Therefore only one set of DOORS login credentials is supported for communication at one time. It is recommended that you use the same DOORS credentials for all configurations so that integration tasks can be performed on the front-end server for all projects at the same time. When a second set of credentials is used, the second set only works after all sessions using of the first set of credentials have timed out.

Rally Integration

Integrating the project management tool Rally enables you to define user stories which you can then use as requirements in Test Manager. You can then create tests in Test Manager to cover these requirements. To update the tests you have created with any changes to the corresponding user stories in Rally, synchronize the requirements.

The following user story properties are synchronized from the Rally REST Service. You can use these properties to help filter and categorize the user stories.:

Property	Description
Name	The name of the user story in Rally.
Rally ID	The identifier of the user story in Rally. This property is the External ID of the requirement in Test Manager.
Description	The description of the user story in Rally.
Iteration	The iteration in Rally in which the user story is included.
Project	The project in Rally in which the user story is included.
Release	The project release in which the user story is included.
State	The state of progress of the user story in Rally. The following states are available: • `Defined` • `In-Progress`

Property	Description
	• `Completed` • `Accepted`
Plan Estimate (Story Points)	The estimated time in Rally for the user story to be completed. The time is estimated in story points.

Enabling Integration with Rally

To enable integration with Rally:

1. In the menu, click **Projects** > **Project List**.
2. Select the project to which you want to establish integration.
3. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
4. Click the **Requirements Management** tab.
5. In the **Rally Integration** section, click **Configure**. The **Edit Configuration** dialog box opens.
6. Type the **URL** of the machine where the Rally REST Service is installed.

   By default, the URL is set to `https://rally1.rallydev.com/slm/webservice/1.22/`.
7. Type valid **Username** and **Password** credentials for the Rally REST Service.
8. *Optional:* If direct access to the Internet is restricted, and the Rally REST service is located in a machine outside the scope of the internal network, use the **Proxy Host** and **Proxy Port** text boxes to specify a proxy through which Test Manager can connect to Rally .

   Note: To connect to Rally by using a proxy, you need to fill out both text boxes, **Proxy Host** and **Proxy Port**.
9. Click **Test Connection** to confirm that the host and user credentials you have entered are correct. If the settings are correct, a **Test connection was successful** dialog box opens.

   Note: If you are not able to establish a connection, consult your system administrator.
10. Click **OK**.
11. Click **Load**. The **Project** list box is populated with all the projects from all workspaces in Rally that you have permissions for.
12. From the **Project** list box, select the external project with which the Test Manager project is to be integrated.
13. *Optional:* Check the **Enable creation of unassigned requirements** check box to enable creation and editing of unmapped requirements in Test Manager projects that are configured for integration with Rally.
14. Click **OK** to save your settings.

Source Control Profile Integrations

Apache Commons Virtual File System

This section describes how to configure Virtual File System (VFS) source control profiles.

A VFS is an abstraction layer on top of a more concrete file system. The purpose of a VFS is to allow client applications to access different types of concrete file systems in a uniform way. Apache Commons VFS provides a single API for accessing various file systems. It presents a uniform view of the files from various sources. The protocols that are currently supported for VFS by Test Manager are:

Protocol	Description
http	Copies the given file. This protocol type is also supported for copying and unpacking ZIP, JAR, or other zipped files. It is required to specify a .zip file on a http server. For example, `zip:http://myTestServer/myTests.zip`. The .zip file will be extracted on the execution server.
ftp	Copies the given file. This protocol type is also supported for copying and unpacking ZIP, JAR, or other zipped files.
smb	Server Message Block (smb) copies all files and folders. This protocol can be used instead of a UNC profile. For example, the VFS smb path `smb://server-name/shared-resource-path` is equivalent to the UNC path `\\server-name\shared-resource-path`.

Note: When you create a new ProcessExecutor test that uses VFS for source control, you need to specify the complete path to the executable in the **Executable Name** text box.

Adding VFS Source Control Profiles

To create a VFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **VFS** from the **Source control system** list box.
6. Type the URL of the VFS **Repository** you want to access.

   Specify the appropriate protocol type in the URL:

FTP	`ftp://<ftp server URL>`
HTTP	http://<http server URL>
SMB	smb://<Samba server url>

   Note: This field is case sensitive.

   Note: HTTP, FTP and SMB are also supported for zipped files. In order to point to a zipped file the URL must be adjusted to `<zipped file type>:<protocol>://<server URL pointing to zipped file>` to include the type of the zipped file. For example, `zip:http://193.80.200.135/<path>/archive.zip` or `jar:http://193.80.200.135/<path>/archive.jar`.
7. Type a valid VFS **Username** and **Password**.

   These credentials will be used to access your VFS repository. The SMB protocol allows including the domain name in the username in the following form: `domain/username`.
8. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
10. Click **OK**.

Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing VFS Source Control Profiles

To modify a VFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the URL of the VFS **Repository** you want to access.

   Note: This field is case sensitive.
6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Edit the **Project path** you want this profile to use.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Concurrent Version System (CVS)

This section describes how to configure CVS source control profiles.

CVS is a powerful source control tool that handles complete source code trees. It can be customized using scripting languages such as PERL and Korn. CVS is decentralized so that users can maintain their own source directory trees. It also enables concurrent file editing.

For a list of the CVS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding CVS Source Control Profiles

To create a CVS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Choose **CVS** from the **Source control system** list box.
6. Type the CVS server name or IP address in the **Hostname** text box.
7. Type the port that is to be connected to in the **Port** text box.
8. Specify the connection method in the **Method** text box.

   Currently, the `ext`, `pserver`, and `local` connection methods are supported.

   This makes the **Port** setting optional.
9. Specify the URL of the CVS **Repository** you want to access.

   For example, `/var/lib/cvs`. If you do not know the URL of the repository, please consult your CVS administrator.
10. Type a valid CVS **Username** and **Password**.

    These credentials will be used to access your CVS repository.

    Note: These settings are optional when using the `ext` connection method.
11. Specify the CVS **Module** that is to be used.
12. Type the **Working folder** to which the Test Manager execution server should copy the source files.

    The working folder must be a local path. For example, `C:\TempSources\`.
13. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
14. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing CVS Source Control Profiles

To modify a CVS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Choose from the following options:

   - Edit the CVS server name or IP address in the **Hostname** text box.

- Edit the port that is to be connected to in the **Port** text box.
- Edit the URL of the CVS **Repository** you want to access. If you do not know the URL of the repository, consult your CVS administrator.
- Edit the CVS **Module** that is to be used.

6. Edit the **Username** and **Password**.

   These credentials are used to access your issue tracking system.

   Note: These settings are optional when using the `ext` connection method.

7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.

8. Specify the connection method in the **Method** text box.

   Currently, the `ext`, `pserver`, and `local` connection methods are supported.

   This makes the **Port** setting optional.

9. Edit the **Project path** you want this profile to use.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Microsoft Team Foundation Server 2010

This section describes how to configure Team Foundation Server (TFS) source control profiles.

For a list of the TFS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding TFS Source Control Profiles

To add a TFS source control profile, the Microsoft Visual Studio Team Explorer Everywhere 2010 command-line client needs to be installed on the front-end server and every execution server, on which you want to use the source control profile. You can download the CLC from the *Microsoft Download Center*.

To create a TFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.

5. Select **Team Foundation Server 2010** from the **Source control system** list box.
6. Enter the **URL** of your TFS installation.

   For example, `http://tfsserver:8080/tfs`.

7. Type the **Domain** of the TFS user.
8. Type a valid **Username** and **Password**.

   These credentials will be used to access your repository.

9. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.

10. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing TFS Source Control Profiles

To add a TFS source control profile, the Microsoft Visual Studio Team Explorer Everywhere 2010 command-line client needs to be installed on the front-end server and every execution server, on which you want to use the source control profile. You can download the CLC from the *Microsoft Download Center*.

To modify a TFS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the URL of the TFS **Repository** you want to access.
6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Edit the **Project path** you want this profile to use.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Microsoft Visual SourceSafe

This section describes how to configure Microsoft Visual SourceSafe (MSVSS) source control profiles.

MSVSS is a version-control system for managing software and Web-site development. Fully integrated with the Visual Basic-, Visual C++-, Visual J++-, Visual InterDev-, and Visual FoxPro development environments, as well as with Microsoft Office applications, MSVSS provides easy-to-use, project-oriented

version control. MSVSS works with any file produced with any development language, authoring tool, or application. Users can work at both the file and project level while promoting file reuse.

For a list of the MSVSS versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding MSVSS Source Control Profiles

Tip: SourceSafe clients must be installed on all front-end, application, and execution servers.

To create a MSVSS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **MSVSS** or **MSVSS (cmd line)** from the **Source control system** list box.

   `MSVSS (cmd line)` utilizes the MSVSS command line plug-in, which works exactly like MSVSS, except that SilkCentral users are automatically logged out of MSVSS when the user logs out from SilkCentral. When selecting `MSVSS`, SilkCentral users remain logged in to MSVSS for an indefinite time.
6. If you selected **MSVSS (cmd line)**, specify the location of the **SourceSafe executable** `ss.exe`.

   SourceSafe must be installed identically on all execution servers and the front-end server. This allows you to specify a definite path. For example, `C:\Program Files\Microsoft Visual Studio\VSS\win32\ss.exe`. If SourceSafe is installed in different locations, see *Configuring the location of a SourceSafe Client*.
7. In the **SourceSafe database (srcsafe.ini)** text box, type the UNC path and file name of the SourceSafe configuration file you want to access or click **Browse** to locate the SourceSafe configuration file..

   Note: SourceSafe configuration files use the name `srcsafe.ini`.
8. Type a valid **UNC username** and **UNC password**.

   These credentials are required to access the UNC path of the configuration file.
9. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
10. Type a valid **Username** and **Password**.

    These credentials will be used to access your repository.
11. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing MSVSS Source Control Profiles

To modify a MSVSS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. If you selected **MSVSS (cmd line)**, specify the location of the **SourceSafe executable** `ss.exe`.

   SourceSafe must be installed identically on all execution servers and the front-end server. This allows you to specify a definite path. For example, `C:\Program Files\Microsoft Visual Studio\VSS\win32\ss.exe`. If SourceSafe is installed in different locations, see *Configuring the location of a SourceSafe Client*.
6. In the **SourceSafe database (srcsafe.ini)** text box, edit the UNC path and file name of the SourceSafe configuration file, or click **Browse** to locate the file.

   If you do not know the location of the configuration file, consult your SourceSafe administrator.

   Note: SourceSafe configuration files use the name `srcsafe.ini`.
7. Edit the **UNC username** and **UNC password**.

   These credentials are required to access your configuration file's UNC path.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
10. Edit the **Project path** you want this profile to use.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Configuring the Location of a SourceSafe Client

To configure the location of a SourceSafe client:

1. In the **SourceSafe executable** text box, type `ss.exe` without any path information.
2. On each execution server and on the front-end server, type the local path of the SourceSafe executable `ss.exe` to the Windows system path. To do this, click **Start** > **Settings** > **Control Panel** > **System** . The **System Properties** dialog box displays.
3. Click the **Advanced** tab and click **Environment Variables**. The **Environment Variables** dialog box displays.
4. Select the **Path** variable in the **System variables** section and click **Edit**.
5. Add the local path of the SourceSafe executable to the list of existing **Variable values**. You can append a new variable value to existing values by entering a semicolon (`;`) followed by the path information.

6. Repeat this procedure for each execution server and for the front-end server.

Serena Version Manager (PVCS)

This section describes how to configure Serena Version Manager (PVCS) source control profiles.

Serena Version Manager, from the makers of PVCS, is the full-featured solution for version control and revision management in software projects. More than simply storing code revisions, Version Manager is a robust, full-featured solution with security, high performance, and varying levels of support for distributed teams. For information on the supported versions, refer to the *Test Manager Release Notes*.

Adding PVCS Source Control Profiles

To create a PVCS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **PVCS** from the **Source control system** list box.
6. Type the UNC path of the PVCS **Repository** you want to access.

   If you do not know the UNC path of the repository, consult your PVCS administrator.
7. Type a valid **UNC username** and **UNC password**.

   These credentials are required to access the UNC path of the configuration file.
8. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Type the **Execution path**. This is the local path of the PVCS installation, where the command line tool `pcli.exe` is located. The default path is `C:\Program Files\Serena\vm\win32\bin`.

   Note: The PVCS client software must be installed on the front-end server and each execution server. PVCS must be installed in identical paths on each machine. For example, if you install PVCS on the TestPartner front-end server to `C:\Program Files\Serena\`, you must install PVCS in the same path on the execution servers.
10. Type a valid **Username** and **Password**.

    These credentials will be used to access your repository.
11. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing PVCS Source Control Profiles

To modify a PVCS source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the UNC path of the PVCS **Repository**. If you do not know the UNC path of the repository, consult your PVCS administrator.
6. Edit the **UNC username** and **UNC password** as required. These credentials are required to access the repository UNC path you specified above.
7. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Edit the **Execution path**. This is the local path of the PVCS installation, where the command line tool `pcli.exe` is located. The default path is `C:\Program Files\Merant\vm\win32\bin`.

   Note: The PVCS client software must be installed on the front-end server and each execution server. PVCS must be installed in identical paths on each machine. For example, if you install PVCS on the Test Manager front-end server to `C:\Program Files\Serena\`, you must install PVCS in the same path on the execution servers.
9. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
10. Edit the **Project path** you want this profile to use.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

SilkTest Workbench

This section describes how to configure SilkTest Workbench source control profiles.

SilkTest Workbench is an automated testing tool that accelerates the functional testing of complex applications. SilkTest Workbench provides support for testing applications developed in a wide variety of development tools including Java, .NET, browser-based web applications, and COM components, including both ActiveX controls and automation objects. With SilkTest Workbench, you can record user sessions with your applications to create tests, enhance the test by adding validation and test logic, and play back tests to ensure that the applications work as expected.

Adding SilkTest Workbench Source Control Profiles

To create a SilkTest Workbench source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.

3. Click **New Profile** to open the **New Issue Tracking Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **SilkTest Workbench Test** from the **Source control system** list box.
6. Select a database type from the **Database type** list box, either SQL Server or Oracle.

   Note: The Microsoft Office Access database type is not supported. When connecting to a SilkTest Workbench Oracle database, there is a SilkTest Workbench requirement that the TNS name must be the same as the host server name.
7. Enter the name of the SilkTest Workbench database server in the **Database server** text box.
8. Enter the name of the SilkTest Workbench database in the **Database name** text box.
9. Enter the port number of the SilkTest Workbench database server in the **Database port** text box.
10. Enter the SilkTest Workbench database schema or owner name in the **Database schema/owner name** text box.
11. Enter the name of the database user in the **Database user** text box.
12. Enter the password for the database user in the **Database password** text box.
13. Enter the name of a valid SilkTest Workbench user in the **SilkTest Workbench User name** text box.
14. Enter the password for the SilkTest Workbench user in the **SilkTest Workbench password** text box.
15. Type the **Working folder** to which the Test Manager execution server should copy the source files.

    The working folder must be a local path. For example, `C:\TempSources\`.
16. Click **Retrieve**. All projects are listed in the **Projects** list.
17. Select one or more projects.
18. Click **OK**.

Editing SilkTest Workbench Source Control Profiles

To modify a SilkTest Workbench source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the SilkTest Workbench **Database type**.

   Note: The Microsoft Office Access database type is not supported.
6. Choose from the following options:

   - Edit the SilkTest Workbench **Database server**.
   - Edit the SilkTest Workbench **Database name**.
   - Edit the **Database port** number of the SilkTest Workbench database server.
   - Edit the SilkTest Workbench **Database schema/owner name**.
   - Edit the SilkTest Workbench **Database user**.
   - Edit the SilkTest Workbench **Database password**.
   - Edit the SilkTest Workbench **User name**.
   - Edit the SilkTest Workbench **Password**.

7. Edit the **Working folder** to which the Test Manager execution server copies the source files.
   The working folder must be a local path. For example, `C:\TempSources\`.
8. Select one or more projects from the **Projects** list.
9. Click **OK**.

StarTeam Source Control Profiles

This section describes how to configure StarTeam source control profiles.

StarTeam promotes team communication and collaboration through centralized control of all project assets. Protected yet flexible access ensures that team members can work whenever and wherever they like through an extensive choice of Web, desktop, IDE, and command-line clients. StarTeam offers a uniquely comprehensive solution that includes integrated requirements management, change management, defect tracking, file versioning, threaded discussions, and project and task management.

For a list of the StarTeam versions that are supported for integration with Test Manager, refer to the *Test Manager Release Notes*.

Adding StarTeam Source Control Profiles

To create a StarTeam source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.
   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **StarTeam** from the **Source control system** list box.
6. Type the **Hostname** of your StarTeam server.
7. Type the **Port** that is to be used to connect to the StarTeam server.
   If the port is not changed, use the default port `49201`.
8. Type a valid **Username** and **Password**.
   These credentials will be used to access your repository.
9. Specify if the profile supports **Encryption**.
10. Type the **Working folder** to which the Test Manager execution server should copy the source files.
    The working folder must be a local path. For example, `C:\TempSources\`.
11. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
12. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing StarTeam Source Control Profiles

To modify a StarTeam source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Choose from the following options:

   - Edit the **Hostname** of your StarTeam server.
   - Edit the port that is to be used to connect to the StarTeam server. If the port is not changed, use the default port `49201`.
   - Specify if the profile supports **Encryption**.
6. Edit the **Project path** you want this profile to use.
7. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Subversion

This section describes how to configure Subversion (SVN) source control profiles.

Subversion (SVN) is the successor to the Concurrent Versions System (CVS). Subversion manages versions using transaction numbers. With each commit, the transaction number is incrementally increased.

What other source control systems call *labels*, Subversion refers to as *tags*. These tags are encoded in the Subversion URL. For example, `http://MyHost/svn/MyApp/trunk` is a Subversion URL and `http://MyHost/svn/MyApp/tags/build1012` is a Subversion tag.

Test Manager supports Subversion tags. If a Subversion URL contains the `trunk` directory, you can define a label `tags/build1012`. This label replaces `trunk` in the Subversion URL.

Note: If a Subversion URL does not contain `trunk` and you define a label, Test Manager throws an error.

Adding Subversion Source Control Profiles

To create a Subversion source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.

3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Choose **Subversion** from the **Source control system** list box.
6. Type the URL of the Subversion **Repository** you want to access.

   If you do not know the URL of the repository, consult your Subversion administrator.

   Note: To use SSH, add *ssh* to the URL, for example `svn+ssh://<hostname>:<port>`.
7. Type a valid **Username** and **Password**.

   These credentials will be used to access your repository.
8. Type a valid **SSH username** and **SSH password** or **SSH keyfile**.

   These credentials are used to access the SSH server. The password overrides the keyfile, so if you only have a keyfile, leave the **SSH password** text box empty. If you use a keyfile, the path to the keyfile must be valid on every execution server that uses the source control profile.
9. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
10. Type the **Project path** you want this profile to use. Click **Browse** to display the **Select Project Path** dialog box. The **Select Project Path** dialog box opens. Select the desired project path in the tree view and click **OK**. Leaving this text box empty sets the project path to the root directory.
11. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing Subversion Source Control Profiles

To modify a Subversion source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **URL** of the Subversion repository you want to access.

   If you do not know the URL of the repository, please consult your Subversion administrator.

   Note: If you cannot check out files, when editing the URL of the Subversion **Repository**, delete the source control mirrors directory on your execution server. For example, `C:\Documents and Settings\All Users\Application Data\Silk\SCC35\SrcCtrlMirrors`.

   Note: To use SSH, add *ssh* to the URL, for example `svn+ssh://<hostname>:<port>`.
6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.

7. Type a valid **SSH username** and **SSH password** or **SSH keyfile**.

   These credentials are used to access the SSH server. The password overrides the keyfile, so if you only have a keyfile, leave the **SSH password** text box empty. If you use a keyfile, the path to the keyfile must be valid on every execution server that uses the source control profile.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Edit the **Project path** you want this profile to use.
10. Click **OK**.

    Test Manager attempts a trial connection to the external system using the information you have provided.

    Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

TestPartner

This section describes how to configure TestPartner source control profiles.

TestPartner is a Micro Focus product that tests Web- and Microsoft Windows-based applications that use Microsoft technologies. TestPartner records user actions to quickly produce powerful tests. Each recorded test displays as a series of actions in clear, concise steps that can be easily understood by all testers, from novice to expert. You can record user sessions with the application, add validation functions, and replay the sessions later to ensure that the application works as expected.

Adding TestPartner Source Control Profiles

To create a TestPartner source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **TestPartner** from the **Source control system** list box.

   Note: When connecting to a TestPartner Oracle database, there is a TestPartner requirement that the TNS name must be the same as the host server name.
6. Select a database type from the **Database type** list box, either SQL Server or Oracle.
7. Type the name of the database server in the **Database server** text box.
8. Type the name of the database in the **Database name** text box.
9. Type the port number of the database server in the **Database port** text box.
10. Type the database schema or owner name in the **Database schema/owner name** text box.
11. Type the name of the database user in the **Database user** text box.
12. Type the password for the database user in the **Database password** text box.
13. Type a valid **Username** and **Password**.

    These credentials will be used to access your repository.
14. Type the **Working folder** to which the Test Manager execution server should copy the source files.

    The working folder must be a local path. For example, `C:\TempSources\`.

15. Use the **Project path** text box to filter which scripts from the database are available to use as tests. Click **Browse** to display the **Select Project Path** dialog box.

    The **Select Project Path** dialog box provides a tree of three levels that you can choose from:

First level	The entire TestPartner database. Note that if you select this level, the **Project path** text box will remain blank.
Second level	The TestPartner project.
Third level	The script type within a specified TestPartner project.

    Select one of the preceeding options and click **OK**. The path will be added to the **Project Path** text box.
16. Click **OK**.

Editing TestPartner Source Control Profiles

To modify a TestPartner source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Choose from the following options:

   - Edit the TestPartner **Database type**.
   - Edit the TestPartner **Database server**.
   - Edit the TestPartner **Database name**.
   - Edit the TestPartner number of the TestPartner database server.
   - Edit the TestPartner **Database schema/owner name**.
   - Edit the TestPartner **Database user**.
   - Edit the TestPartner **Database password**.
6. Edit the **Username** and **Password**.

   These credentials are used to access your repository.
7. Edit the **Project path** you want this profile to use.
8. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
9. Click **OK**.

Universal Naming Convention

This section describes how to configure Universal Naming Convention (UNC) source control profiles.

Short for Universal Naming Convention or Uniform Naming Convention, UNC is a PC format for specifying the location of resources on a local-area network (LAN). UNC uses the following format: `\\server-name \shared-resource-pathname`.

For example, to access the file `test.txt` in the directory `examples` on the shared server `silo`, you would write: `\\silo\examples\test.txt`.

You can also use UNC to identify shared peripheral devices, such as printers. The idea behind UNC is to provide a format so that each shared resource can be identified with a unique address.

UNC is only supported on Microsoft Windows operating systems. If you plan to use a non-windows execution server you can use the Apache Commons VFS source control profile instead.

Adding UNC Source Control Profiles

To create a UNC source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click **New Profile** to open the **New Source Control Profile** dialog box.
4. Type a **Name** for the new profile.

   This is the name that is displayed in lists where the profiles are available for selection.
5. Select **UNC** from the **Source control system** list box.
6. Type the **UNC path** that you want to access.

   This is the path to the location where your test sources are located.
7. Type the **Working folder** to which the Test Manager execution server should copy the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
8. Type a valid **UNC username** and **UNC password**.

   These credentials are required to access the UNC path of the configuration file.
9. Click **OK**.

   Test Manager attempts a trial connection to the external system using the information you have provided.

   Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Editing UNC Source Control Profiles

To modify a UNC source control profile:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Source Control** tab. The **Source Control** page opens, listing all of the source control profiles that have been created for the system.
3. Click on the name of the source control profile that you want to modify. The **Edit Source Control Profile** dialog box opens.
4. Edit the **Name** of the profile.

   This is the name that is displayed in the Test Manager GUI.
5. Edit the **UNC path**.

   This is the path to where your test sources are located.
6. Edit the **Working folder** to which the Test Manager execution server copies the source files.

   The working folder must be a local path. For example, `C:\TempSources\`.
7. Edit the **UNC username** and **UNC password**.

   These credentials are required to access your configuration file's UNC path.
8. Click **OK**.

Test Manager attempts a trial connection to the external system using the information you have provided.

Note: If an error occurs, please review the information that you have supplied, or consult your administrator.

If the trial connection is successful, you are returned to the **Source Control** page.

Test Automation Tools

JUnit Integration

The topics in this section describe the integration between JUnit and Test Manager. You can use the JUnit test type to create Silk4J tests.

Configuring JUnit Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure JUnit test properties:

1. On the **New Test** dialog box, select **JUnit Test** from the **Type** list box and then click **Next**.

   The **JUnit Test Properties** dialog box opens.
2. In the **Test class** text box, type the fully qualified name of the JUnit test class.
3. In the **Test method** text box, type the name of the appropriate test method.

   The method must be available in the test class. If the **Test method** text box is left blank, all tests that are included in the suite will be executed.
4. Set the **Java home directory** to the installation path of the Java Runtime Environment (JRE).

   The path must be valid on the execution server on which the test runs.
5. Specify a valid Java **Classpath** to use on the execution server.

   We recommend to use a relative classpath. The relative classpath is then expanded to the full classpath on the execution server. By using a relative classpath, changes on the location of the source control profile do not require additional changes to the classpath.

   The relative classpath must point to the root node of the test container containing the JUnit test, for example `JUnit_tests`. The relative classpath on the execution server is then expanded to include the working folder of the source control profile, for example `C:\temp`, and the test file names, for example `JUnit4Test.jar`.

   To use a previous version of JUnit, you can also add the relative classpath to `junit.jar` to the classpath, with the appropriate JUnit version. Test Manager supports JUnit 4.4 or later. For example, if the working folder of the source control profile is `C:\temp`, the specified relative paths `junit-4.4.jar;JUnit4Test.jar` are expanded to `C:\temp\JUnit_tests\junit-4.4.jar;C:\temp\JUnit_tests\JUnit4Test.jar` on the execution server. By default, if no relative classpath to the `junit.jar` is specified, Test Manager uses JUnit 4.8.1.

   You can also use a fully qualified classpath. The fully qualified classpath must point to the archive or folder in which the test classes reside. Further, `junit.jar` must be added to the classpath, with the appropriate JUnit version, as the following examples show:

   - `C:\Java\junit4.4\junit.jar;C:\MyApps\main.jar;C:\MyApps\utils.jar`
   - `${junit_home}\junit.jar;${apps_home}\main.jar;${apps_home}\utils.jar`
6. *Optional:* In the **JVM options** text box, you can specify the command-line options and environmental variables that can affect the performance characteristics of the JVM. You can specify multiple options,

but you have to type them in the right order. For example, to use the client VM and set the maximum size of the heap to 512MB, type `-client -Xmx512m`.

7. In the **Coverage path** text box, type the JAR libraries or the specific class files to monitor for code coverage information.

   We recommend using the relative coverage path from the test container root node, which is then expanded on the execution server. You can also use a fully qualified path. Use semicolons to separate multiple jar files, as the following examples show:

   - `C:\MyApps\main.jar;C:\MyApps\utils.jar`
   - `${apps_home}\main.jar;${apps_home}\utils.jar`

   Note: The coverage path setting is disregarded if the **Record external AUT Coverage** check box is checked.

8. Check the **Record external AUT Coverage** check box to get code coverage for the application under test that is defined for the execution plan in the **Code Analysis Settings** portion of **Test Manager** > **Execution Planning** > **Code Analysis** .

   If the check box is not checked, code coverage is recorded from the executing virtual machine. By default, the check box is not checked.

9. Click **Finish**.

Note: Parameters are passed to the Java process as system properties, for example `-Dhost_under_test=10.5.2.133`. Use the `System.getProperty()` method to access the system properties. For example, to access the previously passed `host_under_test property`, use `System.getProperty("host_under_test");`.

Accessing Test Manager Parameters through Java System Properties

Any JUnit test class can access a test parameter of the underlying test as a Java system property; the launcher passes these parameters to the executing machine using the "-D" VM argument.

In addition to the customer-defined parameters , you can always call the following Java system properties from a JUnit test:

Parameter	Description
#sctm_execdef_name	The name of the execution plan that is currently executed. If the execution was triggered from the **Tests** area, the parameter has the value `Unassigned Tests`.
#sctm_execdef_id	The numeric identifier (database key) of the execution plan that is currently executed.
#sctm_product	The name of the product as defined in the executed test container.
#sctm_version	The name of the version to which the results of the execution are associated.
#sctm_build	The name of the version to which the results of the execution are associated.
#sctm_keywords	A comma separated list which contains the keywords that are defined for this execution plan.
#sctm_test_results_dir	The path of the directory where the result files of the test reside.

MSTest Integration

The topic in this section describes the integration between MSTest and Test Manager. You can use the MSTest test type to execute MSTest tests.

Configuring MSTest Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

Note: The MSTest plugin supports the test type unit tests only. Results of other test types are ignored.

To configure MSTest properties:

1. On the **New Test** dialog box, select **MSTest Test** from the **Type** list.
2. Click **Next**. The **MSTest Test Properties** dialog box appears.
3. You can run a test container, a test list, a test class, or a single test method:
   - To run a test container, click **Browse** next to the **Test file** field and select a `.dll` file. You can constrain the run by typing in just a **Test class** or both a **Test class** and a **Test method**.
   - To run a test list, click **Browse** next to the **Test file** field and select a `.vsmdi` file (Visual Studio Test Meta Data). Type a name in the **Test list name** field. You can constrain the run by typing in both a **Test class** and a **Test method**.
4. Click **Finish**.

Note: To run MSTest on an execution server, a Visual Studio distribution or the Visual Studio Test Agent have to be installed. Add the folder that contains `MSTest.exe` to your path variable and restart the execution server. If Visual Studio 2010 is installed, the default path is: `C:\Program Files (x86)\Microsoft Visual Studio 10.0\Common7\IDE`. Currently MSTest distributed with Visual Studio 2010/Test Agent 2010 is supported.

NUnit Integration

The topics in this section describe the integration between NUnit and Test Manager. You can use the NUnit test type to create Silk4NET tests.

Configuring NUnit Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

Note: We recommend that you add the `.\bin` folder of your NUnit installation to the system path. Click **Start** > **Control Panel** > **System** > **Advanced** > **Environment Variables** to add a path like `C:\Program Files\NUnit 2.2\bin` to the system environment variable PATH.

To configure NUnit test properties:

1. On the **New Test** dialog box, select **NUnit Test** from the **Type** list box and then click **Next**.

   The **NUnit Properties** dialog box displays.
2. Click **Browse** to locate and select the NUnit assembly from which you want to pull a test.
3. Type the working directory in the **NUnit Directory** text box.

   This directory is the local path to the file `nunit-console.exe`, such as `C:\Program Files\NUnit 2.2\bin`.
4. In the **NUnit Options** text box, type one or more NUnit console command-line options to specify how NUnit tests are specified.

   For example, to define that the `Flex4TestApp` NUnit test, located in `SilkTest.Ntf.Test.Flex`, should be executed, set the fixture option as follows:

   ```
   /fixture:SilkTest.Flex.Flex4TestApp
   ```

Note: When you add multiple options, you have to separate the options by writing one option in each line in the text box.

5. Click **Finish**.

Process Executor Integration

The topics in this section describe the integration between process executor and Test Manager.

The process executor can be used to launch any executable and extends the published process test launcher class.

Configuring Process Executor Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure process executor test properties:

1. On the **New Test** dialog box, select **ProcessExecutor Test** from the **Type** list box and then click **Next**. The **ProcessExecutor Test Properties** dialog box opens.
2. In the **Executable Name** text box, type the fully qualified name of the executable.
3. In the **Argument List** text box, type all arguments of the process executor test method.

Note: Multiple arguments must be on separate lines in the text box.

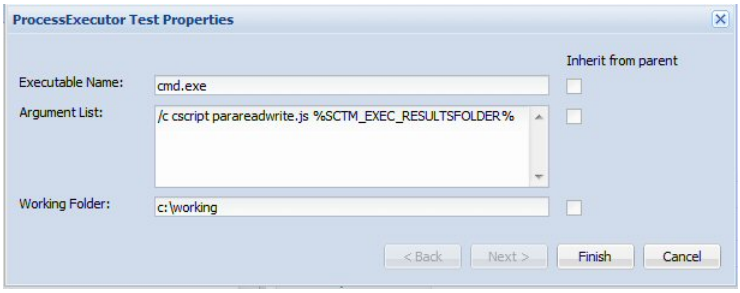
4. Set the **Working Folder**. This is the folder where the executable is executed.

During execution of the executable the following two environment variables can be used:

Environment Variable	Description
SCTM_EXEC_RESULTSFOLDER	All files in this folder are saved as result files in Test Manager. If an `output.xml` file is created during execution in this directory, it is processed by Test Manager.
SCTM_EXEC_SOURCESFOLDER	This is the folder where all source files used during the execution are located.

Example

The following example contains a ProcessExecutor test type that runs the Windows Script Host from the command line with the `/c` switch and two parameters:



- `/c` - command that specifies that the command line should terminate after execution.
- `cscript` - Windows Script Host.
- `parareadwrite.js` - the jscript file named `parareadwrite.js`
- `%SCTM_EXEC_RESULTSFOLDER%` - the Test Manager variable that contains the location of the `Result` folder on the execution server machine. Files stored in this location are automatically uploaded to Test Manager and available in the **Files** tab of the **Test Run**. In this example, the `parareadwrite.js` script produces an

`output.xml` file that is written to the `Result` folder and uploaded back to Test Manager.

SilkPerformer Integration

The topics in this section describe the integration between SilkPerformer and Test Manager.

Configuring SilkPerformer Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure SilkPerformer test properties:

1. On the **New Test** dialog box, select **SilkPerformer Test** from the **Type** list box and then click **Next**.

   The **Test Properties - Select Project** dialog box opens.
2. Perform one of the following steps to define the SilkPerformer project from which your test case is taken:

   - Click **Browse** to select a SilkPerformer project that has been saved to your local file system.
   - Click **Import** to import a SilkPerformer project that is saved to the file pool. A file in the file pool can be used anywhere in the **Tests** tree.
3. On the **Import Project From File Pool** dialog box, either select a saved `.ltz` SilkPerformer project package from the **File pool entry** list box or click **Browse** to select a SilkPerformer project package that has been saved to the source-control system.

   If you check the **Remove file from file pool** check box before you click **Finish**, the selected SilkPerformer file is deleted from the file pool.
4. Perform one of the following steps to upload SilkPerformer projects to the file pool:

   - From **Administration** > **Files**, click **Upload** and browse to the appropriate LTZ file. For more information, refer to the *SilkCentral Administration Module Help*.
   - Use the upload mechanism offered by SilkPerformer.
   - Use the Upload Manager in SilkCentral.
   - Use an existing project directory by way of a UNC path. Create a new test, click **Browse** to select the appropriate LTP file, and then select a workload.
5. On the **Test Properties - Select Project** dialog box, click **Next**.
6. On the **Test Properties - Select Workload** dialog box, select one of the workload profiles that has been defined for the project from the **Workload** list box.
7. Click **Finish** to create the test case.

   Test Manager is fully integrated with SilkPerformer.

Working with SilkPerformer Projects

SilkPerformer is fully integrated with the test and execution functionality of Test Manager. SilkPerformer projects can be integrated into Test Manager tests and directly executed through Test Manager. This allows for powerful test-result analysis and reporting. It also enables unattended testing, which means tests are run automatically by Test Manager based on pre-configured schedules.

Refer to the SilkPerformer Help for details on configuring the integration of SilkPerformer with Test Manager.

SilkPerformer project files can be directly opened in SilkPerformer from Test Manager, where scripts and settings can be edited. Edited SilkPerformer projects can subsequently be checked back into Test Manager to make them available for future test executions.

Test Manager provides information on execution plan run properties during SilkPerformer test executions. Use the `AttributeGet` methods to access execution plan run properties in the SilkPerformer script. You can access the following properties in the script:

- `#sctm_execdef_id`
- `#sctm_execdef_name`
- `#sctm_product`
- `#sctm_version`
- `#sctm_build`
- `#sctm_keywords`

Note: The term Project is used differently in SilkPerformer than it is in Test Manager. A SilkPerformer project, when uploaded to Test Manager, becomes the core element of a Test Manager test. Test Manager projects are high-level entities that may include multiple SilkPerformer projects, tests, execution plans, and requirements.

Downloading SilkPerformer Projects

Whereas opening a SilkPerformer project may involve checking out a SilkPerformer project from a source-control tool, editing the project in SilkPerformer, and checking the project back into Test Manager, *downloading* a project involves downloading a copy of a project and working with it independently of Test Manager. Changes you make to downloaded projects are not automatically migrated back to Test Manager.

To download a SilkPerformer project:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.
4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click . A file download dialog box displays, asking you to confirm that you wish to download the specified SilkPerformer project to your local system.
6. Click **Save** to open the file in SilkPerformer. If not already open in the background, SilkPerformer is invoked.
7. The **Select Target Directory** dialog box displays, loaded with the default directory path to which the specified SilkPerformer project will be saved. If you approve of the specified pathname, click **OK**, otherwise click **Browse** to specify an alternate path.

   Note: Even if you have configured source-control integration, you will not be prompted to check out the SilkPerformer project from your source-control system because you are working with this file independently of Test Manager.

   Note: SilkPerformer projects utilized by Test Manager can also be downloaded directly from the SilkPerformer user interface. For additional information, refer to the SilkPerformer documentation.

Opening SilkPerformer Projects

To open a SilkPerformer project from Test Manager:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.
4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click . A file download dialog box displays, asking you to confirm that you wish to open the specified SilkPerformer command file (.spwbcmd) in SilkPerformer.

6. Click **Open** to open the file in SilkPerformer. If not already open in the background, SilkPerformer is invoked. The **Select Target Directory** dialog box opens, loaded with the default directory path to which the specified SilkPerformer project will be saved.
7. If you approve of the specifed pathname, click **OK**, otherwise click **Browse** to specify an alternate path.
8. If you have configured source-control integration for Test Manager, for example Visual SourceSafe, you are presented with a login screen for your source-control client. Enter valid user connection settings and click **OK** to continue.

   Note: SilkPerformer projects utilized by Test Manager can also be opened directly from SilkPerformer. For additional information, refer to the SilkPerformer documentation.

Executing Attended SilkPerformer Tests

Attended tests are SilkPerformer tests that are executed manually in SilkPerformer and are not executed automatically based on a pre-defined schedule in Test Manager.

Note: To use Test Manager's data-driven test functionality with SilkPerformer scripts, data sources with column names matching the corresponding SilkPerformer project, attributes must be used in conjunction with `AttributeGet` methods.

To execute an attended test run in SilkPerformer:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.
4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click . A file download dialog box displays, asking you to confirm that you wish to run the specified SilkPerformer command file (.spwbcmd).
6. Click **Open** to open the project in SilkPerformer. If not already open in the background, SilkPerformer is invoked. The **Select Target Directory** dialog box displays, loaded with the default directory path to which the specified SilkPerformer project will be saved.
7. If you approve of the specified pathname, click **OK**, otherwise click **Browse** to specify an alternate path. The **SilkPerformer Workload Configuration** dialog box opens with all of the workload settings that are associated with the SilkPerformer project.
8. Edit the workload settings as required and click **Run** to begin the test and monitor the test results with SilkPerformer.

   Note: Clicking **Run** without editing any workload settings executes the SilkPerformer test in exactly the same way as if the test had been executed directly from Test Manager as an unattended test.

Editing SilkPerformer Test Properties

Note: To use the data-driven test functionality of Test Manager with SilkPerformer scripts, you have to use data sources with column names that match the corresponding SilkPerformer project attributes in conjunction with `AttributeGet` methods.

To edit SilkPerformer test properties:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Properties** tab.
4. Scroll down to the **SilkPerformer Test Properties** section.
5. Click **Edit SilkPerformer Test Properties**.
6. Proceed with the configuration of your SilkPerformer test.

Analyzing SilkPerformer Test Results

Performance Explorer enables in-depth analysis of SilkPerformer test results. The **Analyze Results** option downloads only selected results, in contrast to “Downloading result packages”. To assist you in analyzing the results of your optimization efforts, Performance Explorer even allows you to compare statistics from multiple test runs side-by-side in cross load-test reports.

The results of tests that are run using Test Manager can be automatically loaded into Performance Explorer through commands on the **Runs** page in the **Tests** area.

For full details on using Performance Explorer and integrating Performance Manager with Test Manager, refer to the Performance Explorer documentation.

To open SilkPerformer test results in Performance Explorer:

1. In the menu, click **Tests** > **Details View** .
2. Select the test you are interested in viewing.
3. Click the **Runs** tab.
4. Click in the **Actions** column of the test execution for which you want to download results. A **File Download** dialog box displays, showing you the name of the Performance Explorer command file, `.sppecmd`, that you are about to download.
5. Click **Open** to open the results in Performance Explorer. Alternatively, you can click **Save** to save the results locally.
6. If not already open in the background, Performance Explorer now opens, connected directly to your Test Manager installation, and fetches the results of the selected execution run.

   Note: To prepare for a cross load-test report that compares the results of multiple executions in a single report, you may download the results of additional executions from the **Runs** page. Additional execution results are displayed in the existing instance of Performance Explorer on the Performance Explorer Test Manager tab. For additional details on cross load-test reports, refer to the Performance Explorer documentation.

Downloading SilkPerformer Test Result Packages

Downloading result packages is the ideal option if you want to analyze the complete results set of a test run, or if you want to download the complete results set for offline analysis. Because result packages often include large TrueLog On Error files, result packages can be compressed and downloaded to your local hard drive as `.lrz` files. Downloading results locally can also be useful when you are working from a slow Internet connection.

To download SilkPerformer test results:

1. In the menu, click **Tests** > **Details View** .
2. In the **Tests** tree, select a SilkPerformer test.
3. Click the **Runs** tab.
4. Click in the **Actions** column of the test execution for which you want to download results. A **File Download** dialog box displays, showing you the name of the compressed results package file, `.ltz` that you are about to download.
5. Click **Open** to open the results in Performance Explorer. Alternatively, you can click **Save** to save the results locally.
6. If not already open in the background, Performance Explorer now opens. You are presented with an **Import Project** dialog box that indicates the target directory to which the results will be saved. Click **OK** to accept the default path, or click **Browse** to select an alternate path. The downloaded results are then displayed in Performance Explorer.

Note: If you accept the default projects directory where result packages are typically stored, which we generally recommend, then the results will be stored with all otherSilkPerformer results and will be readily accessible through the Performance Explorer `Add Loadtest Results` command.

Uploading SilkPerformer Test Results

Once you have completed running an attended test in SilkPerformer, you can upload the test results to Test Manager and associate the results with a test.

To upload results from an attended SilkPerformer test:

1. Run an attended SilkPerformer test.

   For additional information, see *Executing Attended SilkPerformer Tests*.
2. When the test is complete, select **Upload Results to Test Manager** from the **Results** menu.

   The **Login** screen of the **Upload Results to Test Manager** wizard displays.
3. Enter your **Password** and click **Next**.

   Note: Because this is an attended test, the wizard already knows the appropriate hostname and username of the test to which these results are to be uploaded.
4. If not already selected by default in the project list, select the Test Manager project to which you want to upload the SilkPerformer results.
5. If not already selected by default in the tree list, select the test to which you want to upload the results. Click **Next**.

   Note: You can right-click in the tree and use the commands on the context menu to create a new test, child test, test folder, or child test folder to which the results can be saved.
6. On the subsequent screen you can specify Version and Build numbers for the assigned Product to which the uploaded results belong. Also specify the SilkPerformer test result status, for example `Passed`, `Failed`.

   Note: If any errors occurred during the test run, the test result status is set to `Failed` by default.
7. Click **Finish** to upload the results. The uploaded results are displayed in the **Test Runs** column in **Tests** > **Details View** > **<SilkPerformer Test>** > **Runs**.

Performance Trend Reports

This section explains the performance trend reports that ship with Test Manager. Performance trend reports show the evolution of the application under test's performance over a specified period of time. The input data for the performance reports is provided by SilkPerformer load tests.

Average Page-Time Trend Report

Shows the page times per page for all tests executed for the specified test within the specified time range. The performance trend of the page times for the tested pages is shown in a graph.

Input Parameters

The input parameters for an *Average Page-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.

Parameter	Description
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Measure Filter	Shown measures are limited to those including the specified string in their name. This field has to be filled out. To display all available measures, set the measure filter to `%`. For example, to show only measures that include the word "unit" at any position in their names, set the measure filter to `%unit%`.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Page Time Trend Information

The trend charts show the page time trend over the selected time range for all filtered measures. The minimum, maximum, and average page time curves are shown in each chart. The displayed values in each chart are cut at the selected maximum y-axis value.

Average Transaction Busy-Time Trend Report

Shows the transaction busy time per transaction for all tests executed for the specified test within the specified time range. The performance trends of the transaction busy-times for the tested transaction are displayed in trend charts.

Input Parameters

The input parameters for an *Average Transaction Busy-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Test ID	Identifier of the test for which you want to view the report.

Parameter	Description
Transaction Filter	Shown transactions are limited to those including the specified string in their name. This field has to be filled out. To display all available transactions, set the transaction filter to `%`. For example, to show only transactions that include the word "unit" at any position in their names, set the transaction filter to `%unit%`.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Transaction Busy-Time Trend Information

The trend charts show the transaction busy-time trend over the selected time range for all filtered transactions. The minimum, maximum, and average transaction busy-time curves are shown in each chart. The displayed values in each chart are cut at the selected maximum y-axis value.

Custom Measure Trend Report

Shows the average, minimum, and maximum values of the defined measure or measures for all tests executed for the specified test within the specified time range. The performance trend of the values for each tested measure is shown in a graph.

Input Parameters

The input parameters for a *Custom Measure Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Measures that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Measure Name	Name of the custom measure for which you want to view the report. For example *CreateTestDefinition*.
Measure Type	Type of the custom measure. For example `Transaction(BusyTime)[s]`.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Custom Measure Trend Information

The trend chart shows the performance trend over the selected time range for the selected measure. The minimum, maximum, and average measure curves are shown in the chart. The displayed values in the chart are cut at the selected maximum y-axis value.

Overall Page-Time Trend Report

Shows overall page times, aggregated over all user types, for all tests executed for the specified test within the specified time range. The performance trend of the page times for the tested page is shown in a graph.

Input Parameters

The input parameters for an *Overall Page-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Overall Page-Time Trend Information

The trend chart shows the overall page-time trend over the selected time range for all pages. The minimum, maximum, and average overall page-time curves are shown in the chart. The displayed values in the chart are cut at the selected maximum y-axis value.

Overall Transaction Busy-Time Trend Report

Shows overall transaction busy-time, aggregated over all user types, for all tests executed for the specified test within the specified time range. The performance trend of the transaction busy-times for the tested transaction is displayed in a trend chart.

Input Parameters

The input parameters for an *Overall Transaction Busy-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Overall Transaction Busy-Time Trend Information

The trend chart shows the overall transaction busy-time trend over the selected time range for all transactions. The minimum, maximum, and average transaction busy-time curves are shown in the chart. The displayed values in the chart are cut at the selected maximum y-axis value.

Working with Silk Performance Explorer

Silk Performance Explorer (Performance Explorer) is used for in-depth analysis of test runs. Performance Explorer results analysis can be started directly from the **Execution** area and the **Tests** area of Test Manager through execution runs on the **Runs** page or from Performance Explorer itself. Refer to the Performance Explorer documentation for details regarding the integration of Performance Explorer with Test Manager.

The results of load-test runs in SilkPerformer can also be uploaded to Test Manager and associated with tests. Refer to the *SilkPerformer Help* for more details.

For additional information about the integration of Test Manager integration with SilkPerformer, refer to the *SilkPerformer Help* and the *Performance Explorer User Guide*.

SilkTest Workbench Integration

The topics in this section describe the integration between SilkTest Workbench and Test Manager.

Configuring SilkTest Workbench Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure SilkTest Workbench test properties:

1. On the **Test Properties - Select Scripts** dialog box, select one or more scripts from the **SilkTest Workbench Test** list box.
2. In the **Playback Options** text box, type in the name of a predefined SilkTest Workbench playback option or leave the default value of `System Defaults`.

   Note: A playback option must be set for a SilkTest Workbench test. The delete option on a SilkTest Workbench test will reset the playback option to the default value of `System Defaults`.
3. Click **Finish**.

Opening SilkTest Workbench in the Context of the Result File

1. Click **Execution**.
2. Select your execution plan.
3. Click the **Runs** tab.
4. Click the **Run ID** hyperlink label. The **Test Run Results** dialog box opens.
5. Click the **Files** tab.
6. Click the `result.stwx` file to open SilkTest Workbench in context of the result file.

   Note: SilkTest Workbench must be installed on the machine on which you are trying to open the file.

Viewing the Execution Result Files for a Visual Test

1. Click **Execution**.
2. Select your execution plan.
3. Click the **Runs** tab.
4. Click the **Run ID** hyperlink label.
5. The **Actions** column displays all icons and links for SilkTest Workbench:
   - `detail.htm` - SilkTest Workbench detailed steps file.
   - `result.stwx` - Click to open the result in SilkTest Workbench .
   - `error.png` - Click to go to a screen shot of the last error.

Test Parameters Page - SilkTest Workbench

Tests > **Details View** > **<Test Element>** > **Data Set**

When passing parameters from a Test Manager test to a SilkTest Workbench visual test – the following data types should be used:

Test Manager	SilkTest Workbench
String	Text
Number	Number (Long)
Number	Number (Long Long)
Number	Enumeration
Float	Number (Double)

Test Manager	SilkTest Workbench
Boolean	Boolean (True/False)

SilkTest Classic Integration

The topics in this section describe the integration between SilkTest Classic and Test Manager.

SilkTest Classic Tests

This section describes how to execute tests in SilkTest Classic.

Adding a SilkTest Classic AUT Host

For execution plans that run SilkTest Classic tests, you may have a setup where the SilkTest Classic agent is on a different computer than the execution server. In this case, you can define the location of the SilkTest Classic agent, the SilkTest Classic AUT (Agent Under Test) Hostname.

To add a a SilkTest Classic AUT host to the selected execution plan or edit the host:

1. In the menu, click **Execution Planning** > **Details View**.
2. Select the execution plan to which you want to assign the SilkTest Classic AUT host.
3. Click the **Deployment** tab.
4. In the **SilkTest Classic AUT Hostname** section, click **Edit**. The **Edit SilkTest Classic AUT Hostname** dialog box displays.
5. In the **Hostname** text box, type the name of the computer where the SilkTest Classic agent runs.

   Proper configuration of option files is required. For details on the command-line option `-m`, refer to the SilkTest Classic documentation.
6. Click **OK** to add the SilkTest Classic AUT host to the selected execution plan.

Automated Execution of SilkTest Classic Tests

All tests within an execution plan use the same SilkTest Classic instance for tests. The SilkTest Classic GUI is opened once with the first SilkTest Classic test execution and is closed automatically after the last SilkTest Classic test execution. Each SilkTest Classic test execution produces its own results. If for any reason the SilkTest Classic GUI closes during a test, it will reopen automatically with the next SilkTest Classic test execution.

Automated Execution of Data-Driven SilkTest Classic Testcases

If the **data driven** check box is checked in SilkTest Classic test properties, each SilkTest Classic test will be repeated once for each data row in the external datasource. By default, plan-based execution mode is used for data driven tests. This means that the results of all data rows will be listed under a single node in the `.res` result file. If execution mode is switched to script-based data driven in `SccExecServerBootConf.xml`, a result node will be created in the `.res` result file for each data row.

Specifying Agent Under Test (AUT)

When a SilkTest Classic agent cannot run on the same machine as the Test Manager execution server, for example when tests are run on platforms other than Windows, the hostname and port may be specified by the SilkTest Classic AUT Hostname setting in the **Deployment** page of an execution plan. If the setting has not been defined, SilkTest Classic default values are used, for example from `partner.ini`. The syntax for AUT is `hostname:port`. The agent must be started manually prior to test execution and configured to listen at the specified port. By default, the TCP/IP protocol is used for communication between SilkTest Classic instances and SilkTest Classic agents. Ensure that both programs have been configured to use the same protocol.

Note: Be careful when you have multiple execution servers assigned to an execution plan as SilkTest Classic agents can only work with one SilkTest Classic instance at a time.

SilkTest Classic Time-out Settings

If you have SilkTest Classic test cases that require more than 1 hour to complete, you must adjust the time-out settings in Test Manager. Otherwise, Test Manager assumes that something has gone wrong in the execution and terminates SilkTest Classic. For details about setting the SilkTest Classic time-out, refer to the *SilkCentral Administration Module Help*.

SilkTest Classic Logs

The RMS log file in SilkTest Classic is used to log data for each test case as test runs progress. Three types of data records are written to this file: status, memory and user records. By monitoring this file, the RMS Remote Agent has a means of determining the progress of each test run.

You can write your own comments into the user records of the log file by executing the `PrintToRMSLog` 4Test function.

Examples:

```
PrintToRMSLog ("Error", "An intended error")
PrintToRMSLog ("Info", "testcase sleep1 started")
PrintToRMSLog ("Warning", "TestCase 1 started a second time")
```

Definition of user function in `rms.inc`:

```
PrintToRMSLog (STRING sMessageType, STRING sUserMessage)
```

Writes to the log file in the following format:

```
U|{sTestCaseName}|{sScriptName}|{sArgStr}|{sUserMessage}|{sMessageType}
```

TestPartner Integration

The topics in this section describe the integration between TestPartner and Test Manager.

Configuring TestPartner Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure TestPartner test properties:

1. On the **New Test** dialog box, select **TestPartner Test** from the **Type** list box.
2. Click **Next**. The **Test Properties - Select Test Script** dialog box opens.
3. Click **Browse**.

   The **Choose File** dialog box opens. The assets available in this dialog box are based on the **Project Path** defined in the associated TestPartner source control profile.
4. Select a script from the list and click **OK**.

   Note: To add multiple TestPartner tests, see *Adding Multiple TestPartner Tests*.
5. In the **Playback Options** text box, type in the name of a predefined TestPartner playback option or leave the default value of `System Defaults`.
6. Click **Finish**.

Executing a TestPartner script on a 64-bit machine

1. Download `sqlncli_x64.msi` from *http://www.microsoft.com/downloads/en/details.aspx?familyid=d09c1d60-a13c-4479-9b91-9e8b9d835cdc&displaylang=en*.

2. Install `sqlncli_x64.msi` on the 64-bit box.
3. Go to `C:\Windows\SysWOW64`.
4. Find and Launch `odbcad32.exe` from this directory.
5. Under the **System DSN** tab, verify that the SQL Native Client driver version 2005.90.1399.00 is installed.

Adding Multiple TestPartner Tests

To add multiple TestPartner tests, you must first follow the steps described in *Creating Tests*.

To add multiple TestPartner tests:

1. On the **New Test** dialog box, select **TestPartner Test** from the **Type** list box.
2. Click **Next**. The **Test Properties - Select Test Script** dialog box opens.
3. Click **Browse**.

   The **Choose File** dialog box opens. The assets available in this dialog box are based on the **Project Path** defined in the associated TestPartner source control profile.
4. In the **Playback Options** text box, type in the name of a predefined TestPartner playback option or leave the default value of `System Defaults`.

   Note: The import gives the defined playback option to all imported scripts. If you need to change the playback option for any scripts, edit the test after the import.
5. Click **Next**.

   The **Test Properties - Select Scripts** dialog box opens.
6. In the **TestPartner Scripts** text box, select the scripts to import by clicking `CTRL + Click`.
7. Click **Finish**.

Windows Script Host Tests

Windows Script Host (WSH) is part of the Windows platform and creates an environment for hosting scripts. When a script is to be run at the execution server, WSH plays the role of host. It makes objects and services available for the script and provides a set of guidelines within which the script is executed. Among other things, WSH manages security and invokes the appropriate script engine.

The following online WSH resources might be of value to you:

- *http://labmice.techtarget.com/scripting/WSH.htm*
- *http://www.winguides.com/links.php?guide=scripting*
- Downloadable Script Interpreters: *http://aspn.activestate.com/ASPN/*

Configuring Windows Scripting Test Properties

To configure the properties of a test, you must first follow the steps described in *Creating Tests* or *Editing Tests*.

To configure Windows scripting test properties:

1. On the **New Test** dialog box, select **Windows Scripting Test** from the **Type** list box and then click **Next**.

   The **Windows Scripting Properties** dialog box opens.
2. Click **Browse** and select a Windows scripting test script.
3. Specify the location of any required additional parameters in the **Switches** text box.

Note: You may add other switches to be passed to the script. For more details on the switches that can be used, see *Windows Script Host Tests* and refer to the MS Scripting Host documentation.

4. Click **Finish**.

Supported Script Languages

WSH is language-independent for WSH-compliant scripting engines. Natively, the Windows platform supports Visual Basic Scripts, with file extension `.vbs`, and scripts written in the Java Script language, with file extension `.js`.

For other scripting languages, a dedicated script interpreter must be installed on the execution server. For example, if you install a Perl interpreter on an execution server, this will register a Perl scripting engine at the WSH environment for the extension `.pls`. Whenever a file with extension `.pls` is passed to the WSH tool, with the executable `cscript.exe`, it will invoke the appropriate interpreter because of the file extension. So the client of WSH, in this case the SilkCentral Execution Server, does not need to know about the installation of the Perl interpreter.

Note: After installing a script interpreter, for example Active Perl, try to execute a script locally on the execution server by calling the WSH command line tool with a sample script before executing the script in Test Manager. To do so, open a command shell on the execution server and type `cscript <somescript>`, where `<somescript>` is the path to a script of your choice that is available on your execution server. This is exactly what Test Manager will call when executing a WSH test on an execution server. If the script is executed, then the scripting engine has been registered successfully.

The following scripting languages are WSH compatible:

Scripting Language	File Extension
Perl	`.pls`
Python	`.py`, `.pyw`
REXX	`.REXX`
TCL	`.tcl`

WSH Test Properties

Property	Description	
Script	You can define any file where a script engine is registered for the script language the file contains. Script files under source control are deployed automatically to execution servers, comparable to test sources for other test types.	
Switches	You can enter and pass the following settings to `cscript.exe` during the execution of the test:	
	//B	Batch mode suppresses all non-command-line console UI requests from the script. We recommended that you use this option to prevent a script from waiting for user input during unattended executions at the execution server.
	//U	We recommend to that you use unicode for redirected I/O from the console.
	//T:nn	Time-out, in seconds. The maximum time the script can run, by default = no limit. This option is used to prevent excessive execution of scripts. It sets a timer. When execution time exceeds the specified value, Cscript interrupts the script engine using the `IActiveScript::InterruptThread` method and terminates the process. There is a callback hook. If the time-out is invoked, the `OnTimeOut` function is called to permit cleanup. Although it is possible to create infinite loops using this feature, it is more useful than harmful.

Property	Description	
	//logo	Displays an execution banner at execution time that is visible at the beginning of the `log.txt` log file. This is the default setting.
	//nologo	Prevents display of the execution banner at execution time.
	//D	Enables active debugging.
	//E:engine	Use the engine to execute a script.
	//Job:xxxx	Execute a WSF job.
	//X	Execute the script in debugger.

Using Parameters in WSH Tests

Parameters that are defined for a WSH test automatically add a name and value pair to the command line as an additional argument and set the parameters as environment variables for the called process. This functionality allows you to access all parameters defined for your test within the WSH script.

For example a WSH test is defined with `myscript.js` as script and `//B` as switch. Additionally the test requires a parameter called `IPAddress` with the value `192.168.1.5` and another parameter called `Port` with the value `1492`. The resulting command line for the WSH execution in this example is:

```
csript myscript.js //B IPAddress=192.168.1.5 Port=1492
```

Viewing Information Returned from WSH

To collect results of a WSH execution, the WSH script must generate a file called `output.xml` in the current working directory of the WSH test. All files residing in this directory are stored in the database and are downloadable through the list of files for the test execution. Files are excluded from storage when their extensions are defined under the file extensions to ignore in results property in the **Projects** area.

Note: The current working directory is dynamically created for each WSH execution. Do not use an absolute path when creating the file. Any relative path used will correctly refer to the current working directory.

Any information that a script writes to the WSH standard output goes into the `log.txt` text file that resides in the current working directory. This file is stored in the database and can be viewed as it is included in the file list of the test execution.

The following example shows how to print log information from a script:

```
WScript.Echo "This info will be written to the log.txt file"
```

The XML structure of `output.xml` begins with an element *ResultElement* that defines an attribute named *TestItem*, which specifies the name of the *ResultElement*.

The *ResultElement* must contain an element named *ErrorCount*, optionally an element named *WarningCount*, and a list of Incident elements.

The *ErrorCount* and *WarningCount* elements must contain a positive number or zero. The *ErrorCount* and *WarningCount* of the top-level *ResultElement* are used for evaluating success conditions, which determine if a test has passed or failed. The XML file might contain additional elements that are not visible in the Test Manager GUI. The `output.xml` file is however stored in the database and is viewable as it is included in the file list of the executed test.

The *Incident* element represents an event that happened during the execution of the WSH test. *Message* and *Severity* are shown in the messages list of test executions in the Test Manager GUI. An *Incident* element must contain a *Message* and a *Severity* element.

The *Severity* element must hold one of the following values:

- Info

- Warning
- Error (or Exception)
- Failure

You can store additional information in the result file. The *ResultElement* may contain any number of sub-ResultElements, so information can be easily grouped. Sub-ResultElements make the result file easier to read. For compatibility reasons related to unit tests, JUnit and NUnit, ResultElement can be named TestSuite or Test.

The *ResultElement* may contain the following additional elements:

- *FailureCount*, which is treated the same way as error count
- *RunCount*, if a test is run multiple times
- *Timer*, for example for the duration of the test
- *WasSuccess*, for compatibility with NUnit result files
- *Asserts*, for compatibility with NUnit result files

The *Incident* element may contain a list of *Detail* elements.

The *Detail* element represents detailed information about an *Incident*. It must define a *TestName* element and an *Info* element. The *TestName* is used to give detailed information about where the *Incident* happened. The *Info* element holds detailed information about the *Incident*, for example a stack trace.

Note: Up through Test Manager 8.1, the value of the *Message* and *Info* elements had to be URL encoded (ISO-8859-1). Since version 8.1.1, URL encoding is no longer allowed.

Sample Result File

```
<ResultElement TestItem="WshOutputTest">
  <ErrorCount>1</ErrorCount>
  <WarningCount>1</WarningCount>
  <Incident>
    <Message>some unexpected result</Message>
    <Severity>Error</Severity>
    <Detail>
      <TestName>function main()</TestName>
      <Info>some additional info; eg. stacktrace</Info>
    </Detail>
  </Incident>
  <Incident>
    <Message>some warning message</Message>
    <Severity>Warning</Severity>
    <Detail>
      <TestName>function main()</TestName>
      <Info>some additional info; eg. stacktrace</Info>
    </Detail>
  </Incident>
</ResultElement>
```

Java Script Sample

The following script was used to generate the sample result file. To try this script save it with the extension `.js`.

```
function dumpOutput(dumpFile)
{
  dumpFile.WriteLine("<ResultElement TestItem=\"WshOutputTest
\">");
  dumpFile.WriteLine("  <ErrorCount>1</ErrorCount>");
  dumpFile.WriteLine("  <WarningCount>1</WarningCount>");
    dumpFile.WriteLine("  <Incident>");
```

```
    dumpFile.WriteLine("    <Message>some unexpected result</
Message>");
    dumpFile.WriteLine("    <Severity>Error</Severity>");
    dumpFile.WriteLine("    <Detail>");
    dumpFile.WriteLine("      <TestName>function main()</
TestName>");
    dumpFile.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>");
    dumpFile.WriteLine("    </Detail>");
    dumpFile.WriteLine("  </Incident>");
    dumpFile.WriteLine("  <Incident>");
    dumpFile.WriteLine("    <Message>some warning message</
Message>");
    dumpFile.WriteLine("    <Severity>Warning</Severity>");
    dumpFile.WriteLine("    <Detail>");
    dumpFile.WriteLine("      <TestName>function main()</
TestName>");
    dumpFile.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>");
    dumpFile.WriteLine("    </Detail>");
    dumpFile.WriteLine("  </Incident>");
  dumpFile.WriteLine("</ResultElement>");
}

function main()
{
  var outFile;
  var fso;
  fso = WScript.CreateObject("Scripting.FileSystemObject");
  outFile = fso.CreateTextFile("output.xml", true, true);
  outFile.WriteLine("<?xml version=\"1.0\" encoding=\"UTF-16\"?
>");

  dumpOutput(outFile);
  outFile.Close();
  WScript.Echo("Test is completed");
}

main();
WScript.Quit(0);
```

Visual Basic Script Sample

The following Visual Basic script also generates the sample result file, and saves it as `Output.xml`. To try this script save it with the extension `.vbs`.

```
WScript.Echo "starting"

Dim outFile
Dim errCnt
Dim warningCnt

outFile = "output.xml"
errCnt = 1 ' retrieve that from your test results
warningCnt = 1 ' retrieve that from your test results

Set FSO = CreateObject("Scripting.FileSystemObject")
Set oTX = FSO.OpenTextFile(outFile, 2, True, -1) ' args: file,
8=append/2=overwrite, create, ASCII

oTX.WriteLine("<?xml version=""1.0"" encoding=""UTF-16""?>")
oTX.WriteLine("<ResultElement TestItem=""PerlTest"">")
```

```
oTX.WriteLine("  <ErrorCount>" & errCnt & "</ErrorCount>")
oTX.WriteLine("  <WarningCount>" & warningCnt & "</
WarningCount>")
oTX.WriteLine("  <Incident>")
oTX.WriteLine("    <Message>some unexpected result</Message>")
oTX.WriteLine("    <Severity>Error</Severity>")
oTX.WriteLine("    <Detail>")
oTX.WriteLine("      <TestName>function main()</TestName>")
oTX.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>")
oTX.WriteLine("    </Detail>")
oTX.WriteLine("  </Incident>")
oTX.WriteLine("  <Incident>")
oTX.WriteLine("    <Message>some warning message</Message>")
oTX.WriteLine("    <Severity>Warning</Severity>")
oTX.WriteLine("    <Detail>")
oTX.WriteLine("      <TestName>function main()</TestName>")
oTX.WriteLine("      <Info>some additional info; eg.
stacktrace</Info>")
oTX.WriteLine("    </Detail>")
oTX.WriteLine("  </Incident>")
oTX.WriteLine("</ResultElement>")
```

Communicating with an External System Over SSL

Java Runtime Environment (JRE) version 1.5 or higher must be installed on your computer perform the steps in this task. You can download JRE from *http://java.sun.com/javase/downloads/index.jsp*.

If the certificate of the host you want to connect to is self signed, you may receive the following error message:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target

Note: This error message does not display for valid certificates, which are certificates that are signed by Certificate Authorities.

To use a server with a self-signed certificate, you need to perform the following steps:

1. Download the SSL certificate from the server.

   Use a browser to view the certificate and export it. For example, in Mozilla Firefox, navigate to **Tools** > **Page Info** > **Security** > **View Certificate** > **Details** > **Export**.
2. Start the key- and certificate-management tool *Keytool*.

   Keytool is part of your JRE installation, and is located in `C:\Program Files\Java\jre1.6.0_07\bin`. For additional information on Keytool, see *http://java.sun.com/javase/6/docs/technotes/tools/windows/keytool.html*.
3. To add the certificate to the default Java keystore on the front-end server and application server, type for example the following command in Keytool:

   ```
   keytool
     -importcert
     -file CERTIFICATE.crt
     -keystore "C:\Program Files (x86)\Silk\SC Test Manager 2010\lib\jre\lib
   \security\cacerts"
   ```

   You are prompted to type the password.
4. Type the default keystore password, `changeit`.
5. Restart the front-end server and the application server to reload the keystore.

Reports

Default Reports

Code-Change Impact Reports

The code-change impact reports enable you to perform testing-impact analysis, effort analysis, and risk analysis. You can select classes of interest and, by applying report templates, generate reports that help you determine the test impact that changing the selected classes will result in.

For selected classes, you can choose from report templates that analyze the test impact of proposed code changes.

Code-Change Impact Report for Tests

This report displays the following columns for each affected test:

- Unique key: Test + Execution plan
- Project name
- Test name
- Test hierarchy
- Execution plan
- Test type
- Duration of test
- Status of test, cumulative across all runs of build range
- Last build executed
- # Times executed for this version
- # Times passed for this version + # Times failed for this version
- Coverage index: Methods covered by the test for the specified classes / total methods of specified classes.
- Time stamp
- Test creator
- Test executor. Manual tester or execution server

Code-Change Impact Report for Execution Plans

This report is valuable as it identifies the execution plans that need to be re-run following code changes.

This report displays the following columns for each affected execution plan:

- Project name
- Execution plan name
- # Manual tests
- # Automated tests
- # Manual tests in coverage path
- # Automated tests in coverage path
- Duration of manual tests
- Duration of automated tests
- Duration of manual tests in coverage path
- Duration of automated tests in coverage path

Use Cases for Reports

The following typical code-change impact issues can be addressed with code-change impact reports:

Testing impact analysis	You want to know which tests you should run as a result of a specific change to the code. • Select a particular class. • Select and execute the *Code Change Impact - Tests* report. • Observe the list of tests that cover the classes that were touched in this version.
Effort analysis	You want to know how many hours of automated and manual testing will be required to properly cover a particular set of changes to the code. • Select a particular class . • Select and execute the *Code Change Impact - Execution Plans* report. • Observe the required time (cost) for automated and manual tests.

Code Coverage Reports

This section explains the code coverage reports that ship with Test Manager. Code coverage reports offer a detailed overview of your product's code coverage over a period of time or range of builds.

Code Coverage Trend Report

Shows the improvement trend of code coverage for methods, classes, and packages for a product over a selected range of builds.

Input Parameters

The input parameters for a code coverage trend report are:

Parameter	Description
product_ProductVersion	Version of the selected product.
BuildFrom	First build in the range of examined builds.
BuildTo	Last build in the range of examined builds.

General Report Information

Element	Description
Report Description	Description of the report.
Report Executed By	User who executed the report.
Product Information	Name, version, and examined build range of the selected product.

Code Coverage Trend Graph

Shows the overall percentage of code coverage for the selected product over the selected range of builds. Code coverage for specific packages/namespaces, classes, and methods is displayed individually.

Code Coverage Trend Details

Displays the information from the *Code Coverage Trend Graph* in a tabular format.

Method Coverage Comparison Report

Compares method coverage for all included packages/namespaces across two product builds.

Input Parameters

The input parameters for a method coverage comparison report are:

Parameter	Description
Build 1	Number of the first build that is to be compared.
Build 2	Number of the second build that is to be compared.
Product	The examined product.
Threshold	The minimum amount of change that results in a package/namespace appearing in the report. Packages/namespaces with a smaller percentage of change are not shown in the report. The threshold range is from `0` to `100` percent.

General Report Information

Element	Description
Project Name	Name of the project.
Report Description	Description of the report.
Report Executed By	User who executed the report.
Product	Name of the selected product.
Build 1	Number of the first build to be compared.
Build 2	Number of the second build to be compared.

Method Coverage Information

The method coverage table shows the following information for all packages/namespaces that have changes in method-coverage percentage that are bigger than the threshold:

Element	Description
Package Name	Name of the package.
Statements	Number of statements that are included in the package.
% Method Coverage	Percentage of method coverage in the second build.
% Difference	Difference in the code coverage percentage from the first build to the second build. The difference is negative when code coverage drops.

Context-Sensitive Reports

The areas **Requirements**,**Tests**, and **Executions** offer dynamically-generated lists of reports that are specific to each unit. Context-sensitive report lists are helpful because they offer report types that relate directly to your current activities.

Requirements	Context-sensitive report lists in the **Requirements** tree offer all reports that take a requirement ID as an input parameter.
Tests	Context-sensitive report lists in the **Tests** tree offers all reports that take a test ID as an input parameter.
Executions	Context-sensitive report lists in the **Execution Plans** tree offer all reports that take an execution-plan ID as an input parameter. On the execution-plan **Runs** tab, context-sensitive report lists offer all reports that have the following configuration: • **Result category** = Execution Plan • **Selection criteria** = Execution Plan Run

- **Property** = ID

When you select a report from a context-sensitive report list, you are taken directly to that report's default tab in the **Reports** area. This default destination-tab behavior can be configured using each report's **Edit Report** dialog box.

There are two types of reports that appear in the context-sensitive report lists: reports that you have already accessed and reports that you have not yet accessed. Reports that you have accessed previously appear above a line separator in the menu. These reports are listed chronologically with the most recently viewed report at the top of the list. Other default reports that are available, but have not yet been accessed, appear beneath the line separator.

In addition to the default-configured context-sensitive reports, you can configure new and existing reports to be included in the context-sensitive report list of each area. Context sensitivity is added to reports on a per-user, per-report basis only.

Accessing Context-Sensitive Execution Reports

Note: Reports must be enabled as context-sensitive reports to make them available in the **Executions** area.

To access a context-sensitive execution-plan report:

1. In the menu, click **Execution Planning**.
2. Right-click an execution plan in the **Execution Plans** tree and choose **Reports**.
3. Select a report from the **Reports** sub-menu.

   You are taken to the **Parameters** tab of the selected report in the **Reports** area where the ID of the execution plan is pre-populated as a value.

   Note: You can configure this destination-tab linking behavior by using each report's **Edit Report** dialog box.
4. Edit the parameters of the report as required.
5. Complete configuration of the report on the **Data**, **Report**, or **Chart** page.

Accessing Context-Sensitive Execution-Plan-Run Reports

Note: Reports must be enabled as context-sensitive reports to make them available in the **Executions** area.

To access a context-sensitive execution-plan-run report:

1. In the menu, click **Execution Planning** > **Details View**.
2. Click the **Runs** tab.
3. Right-click a run and choose **Reports**.
4. Select a report from the **Reports** sub-menu.

   You are taken to the **Parameters** page of the selected report in the **Reports** area where the ID of the run is pre-populated as a value.

   Note: You can configure this destination-tab linking behavior by using each report's **Edit Report** dialog box.
5. Edit the parameters of the report as required.
6. Complete configuration of the report on the **Data**, **Report**, or **Chart** page.

Accessing Context-Sensitive Requirements Reports

Note: Context-sensitive reports are available in the **Requirements** area only for those reports that accept a requirement ID as an input parameter.

To access a context-sensitive requirements report:

1. In the menu, click **Requirements** > **Details View** .
2. Right-click a requirement in the **Requirements** tree and choose **Reports**.
3. Select a report from the **Reports** sub-menu.

   You are taken to the **Parameters** page of the selected report in the **Reports** area where the requirement's ID is pre-populated as a value.

   Note: You can configure this destination-tab linking behavior by using each report's **Edit Report** dialog box.
4. Edit the parameters of the report as required.
5. Complete configuration of the report on the **Data**, **Report**, or **Chart** page.

Accessing Context-Sensitive Test Reports

Note: Context-sensitive reports are available in the **Tests** area only for those reports that accept a test ID as an input parameter.

To access a context-sensitive test report:

1. In the menu, click **Tests** > **Details View** .
2. Right-click a test in the **Tests** tree or the **Grid View** and choose **Reports**.

   Note: When multi-selecting tests in the **Grid View**, the context-sensitive reporting is disabled.
3. Select a report from the **Reports** sub-menu.

   You are taken to the **Parameters** page of the selected report in the **Reports** unit where the ID of the test is pre-populated as a value.

   Note: You can configure this destination-tab linking behavior by using each report's **Edit Report** dialog box.
4. Edit the parameters of the report as required.
5. Complete configuration of the report on the **Data**, **Report**, or **Chart** page.

Enabling Context-Sensitive Execution Reports

Explains how to enable execution-plan and execution-plan-run reports to be displayed in context-sensitive report lists.

To enable a simple report to be displayed in context-sensitive report lists in the **Execution** tree or the **Runs** page:

1. Create a new report.

   For additional information, see *Creating New Reports*.
2. Select `Execution Plan` from the **Result category** list box.
3. Select the **Selection criteria** for the context-sensitive report.
4. Select `ID` from the **Property** list box.
5. Type a value in the **Value** text box.

   For example, the ID number of an existing execution plan or an existing execution-plan run.
6. Click **Finish**.

Enabling Advanced Context-Sensitive Execution Reports

To enable an advanced report to be displayed in context-sensitive report lists in the **Execution Plans** tree or on the **Runs** page:

1. Create a report that includes one of the following:
   - An execution-plan ID as an input parameter for the report to be displayed in the **Execution Plans** tree.
   - An execution-plan-run ID as an input parameter for the report to be displayed on the **Runs** page.

   For additional information, see *Creating New Reports* and *Writing Advanced Queries with SQL*.
2. To make an advanced query available in the context menu of the **Executions** area, insert the parameter name `execProp_Id_0` as input for *ExecDef_ID_pk_fk*.

   For example, your report's SQL statement might have defined a hard-coded database-column value, such as `ExecDef_ID_pk_fk = 68`. To edit this report so that it receives column-name values dynamically, replace the static value of `68` with `${execProp_Id_0 | 68}`.

Note: For additional information about tables and column-name definitions, refer to the *Test Manager Database Model Help*.

Enabling Context-Sensitive Requirement Reports

Explains how to enable requirements reports to be displayed in the context-sensitive report list.

To enable a simple report to be displayed in the context-sensitive report list of the **Requirements** area:

1. Create a new report.

   For additional information, see *Creating New Reports*.
2. Select `Requirement` from the **Result category** list box.
3. Select the **Selection criteria** for the context-sensitive report.
4. Type a value in the **Value** text box.

   For example, the ID number of an existing requirement.

Enabling Advanced Context-Sensitive Requirement Reports

To enable an advanced report to be displayed in context-sensitive report lists in the **Requirements** area:

1. Create a report that includes a requirement ID as an input parameter.

   For additional information, see *Creating New Reports* and *Writing Advanced Queries with SQL*.
2. To make an advanced query available in the context menu of the **Requirements** unit, insert the parameter name `reqProp_Id_0` as input for *Req_ID_pk_fk*.

   For example, your report's SQL statement might have defined a hard-coded database-column value, such as `Req_ID_pk_fk = 68`. To edit this report so that it receives column-name values dynamically, replace the static value of `68` with `${reqProp_Id_0 | 68}`.

Note: For additional information about tables and column-name definitions, refer to the *Test Manager Database Model Help*.

Enabling Context-Sensitive Test Reports

Explains how to enable test reports to be displayed in the context-sensitive report list.

To enable a simple report to be displayed in the context-sensitive report list of the **Tests** area:

1. Create a new report.

   For additional information, see *Creating New Reports*.
2. Select `Test` from the **Result category** list box.

3. Select the **Selection criteria** for the context-sensitive report.
4. Type a value in the **Value** text box.
   For example, the ID number of an existing test.

Enabling Advanced Context-Sensitive Test Reports

To enable an advanced report to be displayed in context-sensitive report lists in the **Tests** area:

1. Create a report that includes a test ID as an input parameter.
   For additional information, see *Creating New Reports* and *Writing Advanced Queries with SQL*.
2. To make an advanced query available in the context menu of the **Tests** area, insert the parameter name `tdProp_Id_0` as input for *TestDef_ID_pk_fk*.
   For example, your report's SQL statement might have defined a hard-coded database-column value, such as `TestDef_ID_pk_fk = 68`. To edit this report so that it receives column-name values dynamically, replace the static value of `68` with `${tdProp_Id_0 | 68}`.

Note: For additional information about tables and column-name definitions, refer to the *Test Manager Database Model Help*.

Execution Planning Reports

This section explains the execution planning reports that ship with Test Manager. To ease the assessment of results, execution planning reports give you a detailed overview of the progress of your test executions and the status of defects, over a period of time, or over a range of builds.

Execution Plan Run Comparison Reports

Execution-plan run-comparison reports present an overview of the comparison between two execution-plan runs.

Note: Run-comparison reports are not suitable for the comparison of manual tests to automated tests. When the name of the report includes *(Last Two Runs)*, you can compare only the last two runs of the execution plan or test.

The following reports compare two execution-plan runs:

Report Name	Description
Execution Plan Run Comparison Report	The default execution-plan run-comparison report that compares two execution-plan runs.
Execution Plan Run Comparison Report – Failed in Newer Run	Compares only the failed tests of two execution-plan runs.
Execution Plan Run Comparison Report – Changed Status	Compares only those tests of two execution-plan runs, that changed their statuses.

The execution-plan run-comparison report provides the following information:

- Changes to the status of the execution plans
- Number of errors
- Number of warnings
- Context in which the execution plans were executed
- Execution duration of the assigned tests

Note: When the status of an assigned test changes to `Failed` between compared runs, the test is marked red. When the status of an assigned test changes to `Passed` between compared runs, the test is marked green.

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Execution Plan Run Comparison

This section identifies the following differences between the two runs:

Element	Description
ID	Identifier of the execution plan.
Name	Name of the execution plan.
Start Time	Execution time of each run.
Duration	Duration of each run.
Run ID	ID of each execution plan run.
Product	Name of the product specified for the run.
Version	Version of the product specified for the run.
Build	Build of the product specified for the run.
Status	Status of each execution-plan run.

Test Run Comparison

This section provides the following details about the tests assigned to each execution plan run:

Element	Description
ID	ID of each test.
Name	Name of each test.
Status	Status of each test in each execution-plan run.
Duration	Duration of each test in each execution-plan run.
Errors	Number of errors of each test in each execution-plan run.
Warnings	Number of warnings of each test in each execution-plan run.

Execution Plan Run Errors Report

The *Execution Plan Run Errors* report provides a detailed list of all tests that did not pass an execution-plan run and the reason they did not pass. All errors that occurred during the execution-plan run are listed in this report. The user can quickly assess results and easily identify any unwanted effects in the execution-plan run.

Input Parameters

The input parameter for an **Execution Plan Run Errors** report is the identifier of the execution-plan run.

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Execution Plan Information

This section provides the following information about the execution plan:

Element	Description
Execution Plan ID	Identifier of the execution plan.
Execution Plan Name	Name of the execution plan.
Run ID	Identifier of the execution-plan run.
Product	Name of the product specified for the execution-plan run.
Version	Version of the product specified for the execution-plan run.
Build	Build of the product specified for the execution-plan run.
Execution Server	Execution server where the execution plan was run.
Keywords	Keywords assigned to the execution-plan run.
Execution Timestamp	Time and date of the execution-plan run.
Duration	Duration of the execution-plan run.
Status	Status of all tests assigned to the execution plan.

Test Runs

This section provides the following information about each test run that did not pass:

Element	Description
ID	Identifier of the test.
Name	Name of the test.
Duration	Duration of the test run.
Errors	Amount of errors that occurred during the test run.
Warnings	Amount of warnings that occurred during the test run.
Messages	If there are messages available, the content of the **Messages** page of the **Test Run Dialog** is shown here.

Not Passed Tests Report

The *Not Passed Tests Report* presents an overview of all not-passed tests of a folder or project in the **Executions** area. You can use this report for error analysis, for example during configuration testing.

Input Parameters

The input parameter for a *Not Passed Tests Report* is the identifier of the configuration suite, folder, or project.

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Specific Information for Each Test

For each test, the report provides a grid with an entry for each execution plan in context of which the test is not passed. Click on the name of the test to view the **Runs** page of the test in the **Tests** area. The grid includes the following columns:

Column	Description
ID	Identifier of the execution plan.
Execution Plan	Name of the execution plan. Click to view the runs of the execution plan in the **Executions** area.
Status	The status of the last run of the test in the context of the execution plan. For data-driven tests or test packages, the status is displayed as a bar graph.
Run ID	Identifier of the last run of the test. Click to view the results of the run in the **Test Run Results** dialog box.

Not Passed Tests per Execution Plan Report

The *Not Passed Tests per Execution Plan Report* presents an overview of all not-passed tests per configuration of a configuration suite, or per execution plan of a folder or project in the **Executions** area. You can use this report for error analysis, for example during configuration testing.

Input Parameters

The input parameter for a *Not Passed Tests per Execution Plan Report* is the identifier of the configuration suite, folder, or project.

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.

Element	Description
Report Executed By	User who executed the report.

Specific Information for Each Execution Plan

For each configuration or execution plan, the report provides a grid with an entry for each test in the configuration or execution plan. Click on the name of the configuration or execution plan to view the **Properties** page of the configuration or execution in the **Execution** unit. The grid includes the following columns :

Column	Description
ID	Identifier of the test.
Test	Name of the test. Click to view the runs of the test in the **Tests** area.
Status	The status of the last run of the test in the context of the execution plan. For data-driven tests or test packages, the status is displayed as a bar graph.
Run ID	Identifier of the last run of the test. Click to view the results of the run in the **Test Run Results** dialog box.

Execution Status Overview Report

The *Execution Status Overview* report presents an overview of the execution status of the selected node and all execution plans below the node.

Input Parameters

The input parameter for a *Execution Status Overview* report is the identifier of the node in the **Execution Plans** tree.

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the project that includes the selected node.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Overall Status Information

Provides an overview, including the aggeregated status, of the selected node and all execution plans below the node.

Element	Description
Name	Name of the selected **Execution** tree node.
Execution Plans	Amount of execution plans that are that are below the selected node and were executed.
Tests	Amount of tests that are assigned to the execution plans below the selected node.
Status	The aggregated status of the last run of the execution plans below the selected node. The status is displayed as a bar graph.

Element	Description
Duration	Accumulated duration of the last run of the execution plans below the selected node.

Status of Execution Plans

Provides detailed information on the execution status of the individual execution plans, including the selected node, that are below the selected node. Can include multiple tables, if the selection includes multiple parent tree nodes.

Column	Description
Name	Name of the execution plan. Click to view the properties of the execution plan in the **Executions** area.
Tests	Amount of tests that are assigned to the execution plan.
Status	The status of the last run of the execution plan.
Product	Name of the product that the execution plan is assigned to.
Version	Version of the product that the execution plan is assigned to.
Build	Build of the product that the execution plan is assigned to.
Start Time	Date and time the last run of the execution plan started.
Duration	Duration of the last run of the execution plan.

Issues Per Component Report

The *Issues Per Component report* offers an overview of all issues related to each component. In addition to offering basic issue tracking, this report assists in monitoring the overall issue trend for each component.

Performance Trend Reports

This section explains the performance trend reports that ship with Test Manager. Performance trend reports show the evolution of the application under test's performance over a specified period of time. The input data for the performance reports is provided by SilkPerformer load tests.

Average Page-Time Trend Report

Shows the page times per page for all tests executed for the specified test within the specified time range. The performance trend of the page times for the tested pages is shown in a graph.

Input Parameters

The input parameters for an *Average Page-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.

Parameter	Description
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Measure Filter	Shown measures are limited to those including the specified string in their name. This field has to be filled out. To display all available measures, set the measure filter to `%`. For example, to show only measures that include the word "unit" at any position in their names, set the measure filter to `%unit%`.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Page Time Trend Information

The trend charts show the page time trend over the selected time range for all filtered measures. The minimum, maximum, and average page time curves are shown in each chart. The displayed values in each chart are cut at the selected maximum y-axis value.

Average Transaction Busy-Time Trend Report

Shows the transaction busy time per transaction for all tests executed for the specified test within the specified time range. The performance trends of the transaction busy-times for the tested transaction are displayed in trend charts.

Input Parameters

The input parameters for an *Average Transaction Busy-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Test ID	Identifier of the test for which you want to view the report.
Transaction Filter	Shown transactions are limited to those including the specified string in their name. This field has to be filled out. To display all available transactions, set

Parameter	Description
	the transaction filter to %. For example, to show only transactions that include the word "unit" at any position in their names, set the transaction filter to %unit%.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Transaction Busy-Time Trend Information

The trend charts show the transaction busy-time trend over the selected time range for all filtered transactions. The minimum, maximum, and average transaction busy-time curves are shown in each chart. The displayed values in each chart are cut at the selected maximum y-axis value.

Custom Measure Trend Report

Shows the average, minimum, and maximum values of the defined measure or measures for all tests executed for the specified test within the specified time range. The performance trend of the values for each tested measure is shown in a graph.

Input Parameters

The input parameters for a *Custom Measure Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Measures that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Measure Name	Name of the custom measure for which you want to view the report. For example *CreateTestDefinition*.
Measure Type	Type of the custom measure. For example `Transaction(BusyTime)[s]`.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Custom Measure Trend Information

The trend chart shows the performance trend over the selected time range for the selected measure. The minimum, maximum, and average measure curves are shown in the chart. The displayed values in the chart are cut at the selected maximum y-axis value.

Overall Page-Time Trend Report

Shows overall page times, aggregated over all user types, for all tests executed for the specified test within the specified time range. The performance trend of the page times for the tested page is shown in a graph.

Input Parameters

The input parameters for an *Overall Page-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Overall Page-Time Trend Information

The trend chart shows the overall page-time trend over the selected time range for all pages. The minimum, maximum, and average overall page-time curves are shown in the chart. The displayed values in the chart are cut at the selected maximum y-axis value.

Overall Transaction Busy-Time Trend Report

Shows overall transaction busy-time, aggregated over all user types, for all tests executed for the specified test within the specified time range. The performance trend of the transaction busy-times for the tested transaction is displayed in a trend chart.

Input Parameters

The input parameters for an *Overall Transaction Busy-Time Trend* report are:

Parameter	Description
Date From (DD-MON-YYYY)	Starting date for the time range. For example `06-DEZ-2008`.
Date To (DD-MON-YYYY)	End date for the time range. For example `16-JAN-2009`.
Exclude Runs with more than <nnn> Errors	Runs that generate more errors than specified here are not included in the report. Use this setting to avoid that outliers skew the trend curve.
Maximum Value for y-Axis	Limits the y-axis of the graph to the specified value. Transaction busy-times that exceed this value are cut off at the top. This setting is useful to prevent the flattening of lines caused by outliers.
Test ID	Identifier of the test for which you want to view the report.

General Report Information

Lists overview information like the name of the current project, the report description, and the user who executed the report.

Test Information

Lists general information about the test.

SilkPerformer Project Information

Lists general information about the SilkPerformer project that is used to perform the load test.

Overall Transaction Busy-Time Trend Information

The trend chart shows the overall transaction busy-time trend over the selected time range for all transactions. The minimum, maximum, and average transaction busy-time curves are shown in the chart. The displayed values in the chart are cut at the selected maximum y-axis value.

Project Overview Report

Test Manager > **Tracking** > **Project Overview Report**

The *Project Overview Report* contains a high-level overview of the status of the selected project.

Requirements Reports

This section describes the requirements-related reports that ship with Test Manager. Requirements reports detail the status of functional requirements, for example compatibility requirements, GUI requirements, or feature requirements, which must be met during development. Requirements may also relate to product management objectives such as reliability, scalability, and performance. The requirement-management reports help managers to determine if adequate test coverage is established to verify that system requirements are met during development. When a report references a requirement that includes HTML-formatted content, that content is rendered in the report.

Status Reports

The following status reports are available for the **Requirements** area:

Report	Description
Requirements Status Overview	Represents a grouped summary of all requirements by current requirement coverage. Coverage is expressed by the statuses `Passed`, `Failed`, `Not Executed`, and `Not Covered`.
Top-Level Requirement Coverage	Represents a listing of all top-level requirements. For each requirement the number of child requirements, that are covered and not-covered by tests, is displayed.
Status of Requirements with Priority Higher than 'X'	Represents a summary of all requirements by current requirement coverage. The returned group of requirements is restricted by the `Priority` parameter, which specifies the lowest requirement priority that is considered in the data.
Requirement Impact Analysis Report	Displays the test details grouped by the execution plan that is associated to the requirement. This report allows the user to gain insight into testing assets that may be impacted by a change to the requirement. The report contains the name, status, date/time of last execution, manual testers, planned time, and issues of each test. The following statuses are used: **Status** — **Description** **Passed** — The test passed. **Failed** — The test failed. **N/A** — The test is associated to a requirement but is not submitted for execution. **Not Executed** — The test is associated to a requirement and is submitted for execution but not executed yet.
Project Requirement Traceability Report	Displays all requirements in a project along with their associated tests and issues. The following statuses are used: **Status** — **Description** **Passed** — The test passed. **Failed** — The test failed. **N/A** — The test is associated to a requirement but is not submitted for execution. **Not Executed** — The test is associated to a requirement and is submitted for execution but not executed yet.

Progress Reports

The following progress reports are available for the **Requirements** area:

Report	Description
Requirements Coverage Across Builds 'X' and 'Y'	Represents a summary of all requirements by current requirement coverage. The returned group of requirements is restricted by the `Priority` parameter, which specifies the lowest requirement priority that is considered in the data.

Report	Description
Requirements Coverage Over the Past 'X' Days	Represents a trend in requirements coverage by considering overall requirements coverage over 'X' days.
Specific Requirements Coverage Over the Past 'X' Days	Represents a trend in requirements coverage by considering specific requirements coverage over 'X' days.

Document Reports

The following document reports are available for requirements:

Report	Description
All Requirements	All requirements are represented with full requirement information.
Requirement with Child Requirements	The selected requirement is shown with its requirement ID. Full details regarding the child requirements of the requirement are displayed.

All Related Issues Report

The *All Related Issues* report provides a detailed list of all issues related to the assigned tests for a requirement, and explains the relationship between requirements, the assigned tests, and issues that have occurred.

Input Parameters

The input parameter for the *All Related Issues* report is the identifier of the requirement.

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Requirement Information

This section provides the following information about the requirement:

Element	Description
ID	Identifier of the requirement.
Name	Name of the requirement.
Description	Description of the requirement.
Nr. of Issues	Amount of issues related to the requirement or sub-requirements of the requirement.

Related Issues

Shows all issues related to the requirement or sub-requirements of the requirement in tabular form. The table provides the following information for each issue:

Column	Description
ID	Identifier of the issue. If an identifier is provided by the issue tracking system, this external identifier is used. The identifier is clickable if an external link is defined for the issue.
Synopsis	Meaningful short-description of the issue.
Status	Current status of the issue. If the status is provided by the issue tracking system, this external status is used.
Assigned by	Person who assigned the issue to the test.
Test ID	Identifier of the test in which the issue was discovered.
Test	Name of the test in which the issue was discovered.

Test Reports

This section explains the test-related reports that ship with Test Manager. Test reports give you an overview of the progress of your tests and the status of defects over a period of time or over a range of builds.

Status Reports

The following status reports are available for the **Tests** area:

Report	Description
Test Status Overview	Represents a status overview of all tests, structured by the statuses `Passed`, `Failed`, `Not Executed`, and `N/A`.
Test Status Overview (per test container)	Represents a status overview of all tests contained in a specific test container, structured by the statuses `Passed`, `Failed`, `Not Executed`, and `N/A`.
Tests per Component	Represents an overview of coverage of components by test; makes it easier to see where testing activity is needed.
Tests (per test container)	Represents a success rate for each test container by listing the number of passed tests.
Implemented Tests (per component)	Represents an overview of coverage of components by tests that have the *Implemented* attribute set to `Yes`.
Failed Tests (per component)	Represents an overview of failed tests per component; makes it easier to identify the most critical components in the environment.

Quality Goals Planning Report

Reports > **Details View** > **Tests** > **Status Report**

Represents the number of tests in each goal criteria along with the planned time rolled up to the goal criteria.

Note: The planned time value is only for manual tests. Automated tests do not have planned time.

This report is split three ways for each quality goal defined in a project:

- **Testing Scope** - compares the total number of tests available against the total number or required tests for each item in a quality goal.
- **Required Planned Time** - compares the required planned time among the items in the quality goals.
- **Value table** - lists all of the quality goal items along with related metrics: **Tests Available**, **Goal (%)**, **Tests Required**, and **Planned Time for Manual Tests**.

Note: For a multi select attribute/property, the test will be counted for each matching occurrence.

Quality Goals Execution Report

Reports > **Details View** > **Tests** > **Status Report**

Visually displays the number of tests executed for each item of each goal. This report is split into two sections for each quality goal defined in a project:

- **Execution Status Graph** - lists each quality goal item with a color-coded execution status and summary value.
- **Value table** - lists all of the quality goal items as well as their execution metrics: **Passed**, **Failed**, **Not Executed**, **N/A**, **Total**, **Test to Meet Goal**, and **Execution % to Goal**.

Progress Reports

The following progress reports are available for the **Tests** area:

Report	Description
Test Progress Across Builds 'X' and 'Y'	Represents a trend in test progress resulting from viewing test statuses in context with builds. The user must specify a build range consisting of a start- and an end-build.
Manual Test Coverage since Build <x> (cumulative)	Represents a trend in manual test progress, excluding all kind of automated tests, resulting from viewing test statuses in context with builds. The user must specify a starting build number which is used as basis for calculating a cumulative progress.
Automated Test Coverage for Builds (non-cumulative)	Represents the automated test coverage resulting from viewing test statuses in context with builds. The user must specify a version and a product and the report shows the test coverage that could be achieved with the automated tests per build.
Test Progress this Month	Represents a trend in test progress resulting from viewing test statuses for the current month.
Specific Test Node Progress Over the Past 'X' Days	Represents a trend in requirements coverage by considering a specific test node over the past 'X' days.
Tests Created in the Past 'X' Days (per component)	Represents a listing of new tests per component over the past 'X' days. Assists in identifying components that lack testing activity.
Test Progress Over the Past 'X' Days	Represents a trend in test progress by considering test statuses over the past 'X' days.
Percentage Testing Success Over the Past 'X' Days (per component)	Represents a percentage listing of successful tests per component over the past 'X' days; assists in identifying the components in the environment that are most critical.

Manual Test Reports

The following manual-test reports are available for the **Tests** area:

Report	Description
Planned vs. Actual Execution Time of Manual Tests (Summary)	Represents an overview of the deviation between planned and actual time for execution of manual tests, viewed on a daily basis.
Historic Planned vs. Actual Execution Time (per user)	Represents an overview of planned and actually required execution time for completed manual tests per user over a specific period of time.

Report	Description
Planned vs. Actual Execution Time (status per user)	Represents progress in terms of planned vs. actual hours of currently pending manual tests per user. Manual tests are only considered if test results have been entered by the user and are assigned to the user who enters the results.
Manual Test Result Document	An easily printable manual test case result report for a single test.
Manual Test Document	An easily printable manual test case report for a single test.

Baseline Comparison Report

Reports > **Details View** > **<Active Project>** > **Test** > **Baseline Comparison** > **Baseline Comparison**

The *Baseline Comparison* report compares a baselined project with the baseline and displays the number of changed, deleted, and created tests.

Input Parameters

The input parameters for the *Baseline Comparison* report are the identifiers of the project and the baseline.

General Report Information

This section provides a description of the report and the name of the user who executed the report.

Project and Baseline Information

This section provides the information on the project and the baseline, including data related to the creation of the baseline.

Project and Baseline Summary

This section provides a sum of the created, changed, and deleted tests in the baseline and the project, in both tabular and graphical form.

Changes

This section provides a detailed list of all changed tests in the project and the baseline, with links to the tests.

Test Run Comparison Report

Test run-comparison reports present an overview of the comparison between two runs of a test.

Note: Run-comparison reports are not suitable for the comparison of manual tests to automated tests. When the name of the report includes *(Last Two Runs)*, you can compare only the last two runs of the execution plan or test.

The test run comparison report provides the following information:

- Changes to the status of the test
- Number of errors
- Number of warnings
- Context in which the test was executed
- Execution duration of the assigned tests
- Attributes and properties of the test
- Parameters of the test
- Success conditions for the test

General Report Information

This section provides the following general information about the report:

Element	Description
Project Name	Name of the active project.
Report Description	Description of the report.
Report Executed By	User who executed the report.

Test Information

This section provides the following information about the test:

Element	Description
ID	Identifier of the test.
Name	Name of the test.
Description	Description of the test.

Execution Information

This section provides the following information about each execution:

Element	Description
Execution ID	ID of each execution plan.
Execution Name	Name of each execution plan.
Run ID	ID of each execution plan run.
Product	Name of the product.
Version	Version of the product.
Build	Build of the product.

Test Run Comparison

This section identifies the following differences between the two runs:

Element	Description
Status	Status of each run.
Execution Timestamp	Timestamp of each run.
Duration	Duration of each run.
Errors	Number of errors in each test run.
Warnings	Number of warnings in each test run.
Previous Status	Status of each run previous to the last manual change.
Changed by	User who performed the last manual change to the status.
Change Comment	Describes the reason of the manual status change.

Attributes and Properties

This section identifies the attributes and properties of the two runs of the test at execution time.

Parameters

This section lists the parameters of the two runs of the test at execution time.

Success Conditions

This section lists the conditions at execution time for each of the two runs to be considered successful. If a condition is not satisfied, the test run is considered unsuccessful. Satisfied conditions are marked green, while unsatisfied conditions are marked red.

Creating and Customizing Reports

Creating Reports

Test Manager offers reports that quickly and easily transform data into intuitive charts and graphs. Pre-installed reports are available for the **Requirements**, **Tests**, and **Issues** areas.

Reports are created using either BIRT RCP Designer, an open-source, Eclipse-based report tool, or Microsoft® Excel® report templates. Test Manager is tightly integrated with BIRT RCP Designer to make it easy for you to generate reports on test management data. The reporting functionality in Test Manager is highly customizable. Numerous pre-installed reports and report templates provide out-of-the-box options for a wide range of reporting needs. Simple GUI-based tools allow you to edit the pre-installed reports and create reports of your own. For users with SQL knowledge, there is virtually no limit to how data can be queried and presented in custom reports.

Tip: If a blank report is generated, the cause may be that there are not any data in the project you selected, or you may not be connected to the appropriate Test Manager database. Reports are not available offline unless your Test Manager database is accessible locally.

Sample Report

Below is the code of a pre-installed report called *All Requirements*. This report has not undergone editing using the GUI-based tools of Test Manager or SQL. By default, this report displays all properties of all requirements in the selected project, except those requirements that have been identified as obsolete. Obsolete requirements are filtered out by the report's `reqProp_Obsolete_0` parameter.

```
SELECT r.ReqID, r.ReqParentID, r.PositionNumber, r.ProjectID, r.ProjectName,
r.ReqName,
r.Risk, r.Priority, r.ReqDescription, r.ReqCreator, r.ReqCreated,
r.ReqReviewed, r.ReqCoverageStatus,
r.ReqRevision, r.MarkedAsObsolete, r.Obsolete, r.TreeOrder
FROM RTM_V_Requirements r
WHERE r.ReqID IN (SELECT DISTINCT ReqTreeNodeID_pk as id
FROM TM_RequirementTreeNodes rtn  WITH (NOLOCK)
WHERE rtn.ProjectID_fk = 98
AND  rtn.MarkedForDeletion=${reqProp_Obsolete_0|0}
AND ParentTreeNodeID_fk IS NOT NULL)
```

Creating New Reports

To create a new report:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the folder in which you want the new report to display.
   This determines where the report is stored in the directory tree.

3. Click on the toolbar. The **Create New Report** dialog box opens.
4. Type the name of the new report.

   This is the name that is displayed in the **Reports** tree.
5. Check the **Share this report with other users** check box if you want to make this report available to other users.
6. In the **Timeout [s]** field, type the maximum time period in seconds that Test Manager should wait for SQL queries to complete.
7. From the **Default tab** list box, select the tab that you want to be directed to when you select this report from one of the context-sensitive report lists.
8. Select the corresponding result type from the **Result category** list box.

   This setting specifies the database table and view that is to be filtered for the report. The following result types are available:

Result Type	Description
Requirement	Returns requirements available in the **Requirements** module that meet the query criteria.
Test	Returns tests available in the **Tests** area that meet the query criteria.
Test Execution	Returns executed test results from the **Executions** area that meet the query criteria.
Execution Plan	Returns execution plans from the execution area.
Issue	Returns issues, including imported issues.
Requirement Progress Builds	Contains information on requirements progress per build so that you can see how requirements develop across builds.
Requirement Progress Days	The same as Requirement Progress Builds, but shows development on a daily basis.
Test Progress Builds	Shows how tests develop across builds.
Test Progress Days	Same as above, but shows development on a daily basis.

   Each result type offers a set of selection criteria. Based on the result type you have selected, specify an appropriate **Selection criteria** for your report. These criteria typically group properties based on a view or some other intuitive grouping, for example custom properties.
9. From the **Property** list box, select the property that is to be filtered on.

   For some selection criteria, properties are dynamic.
10. Select an **Operator** for the query.

    The available operators depend on the property. Example operators are `=`, `not`, `like`, and `not like`. Strings are always compared lowercase. Allowed wildcards for strings are "*" and "?", where * matches any characters and ? matches exactly one character.
11. Select or specify the **Value** that the query is to be filtered on.

    For date-based properties, the **Value** field is replaced with a calendar tool that you can use to select a specific date.
12. *Optional:* To add an additional query string to this report, click **More**. An existing query string can be deleted by clicking the string's **Delete** button. When multiple query strings are defined, **AND** and **OR** option buttons are displayed next to **More**. Use these option buttons to define if the queries should be considered cumulatively, or if only one query string's criteria needs to be met.
13. Click **Next** to configure report columns on the **New Report** dialog box.
14. Click **Add Columns**. The **Add Columns** dialog box lists all available report columns.
15. Select the columns that you want to have included in the report and click **OK**.

    You can select multiple columns with **CTRL + CLICK**.

Note: For test-planning reports, the list of available column names is enhanced with the column names from the *LQM_v_tests* table. For details, refer to the *Test Manager Database Model Help*.

The selected columns display in tabular format on the **New Report** dialog box.

16. *Optional:* Configure how each report column is to be displayed. For each column, specify a sort direction, `ascending`, `descending`, or `unsorted`, using the up/down arrows in the **Sorting** column.
17. When multiple columns are selected for sorting, a list box is displayed in the **Sort Order** column that allows you to more easily edit the column-sort order. Set these numbers as required.
18. Give each column an **Alias**.

    This is the name by which each column will be labeled in the generated report.
19. With grouping, you can take advantage of SQL aggregation features, for example when selecting a number of elements or querying a total sum of values. Check the **Group by** check box to specify that SQL group by functions are to be applied.
20. Columns that are not selected for SQL group by functions are set to aggregation by default, which means a single aggregate value is calculated. From the **Aggregation** list box, select the appropriate aggregation type.

    The following types are available:

    - Count
    - Sum
    - Average
    - Minimum
    - Maximum
21. The **Actions** column enables you to move column listings up and down in the view, or to delete a column. The **Move Up** and **Move Down** functions do not affect the outcome of the report.
22. Click **Finish** to complete your new report.

SQL Functions for Custom Reports

To assist in writing advanced queries, placeholders are available for each function. Function placeholders are replaced with SQL code upon execution. Functions are used like parameters, but their names have a $ (dollar symbol) as a prefix. Unlike parameters, placeholders are defined report elements that cannot be customized per execution.

The following table lists all available function placeholders:

Function	What it does	Example
$TODAY	Returns the current systemdate on the database server. You can also write `$TODAY-1` for yesterday or `$TODAY-7` for a week ago.	`CreatedAt > ${$TODAY}`
$DATE(column)	Returns the date but not the time.	
$DATE('string')	Converts the given string to a database date.	`CreatedAt > ${$DATE('01/10/2005')}`
$DAYS[p1;p2]	Calculates the difference in days between the two given parameters. The two parameters can be a column within the table/view or `$TODAY`.	The following example returns the rows created within the last week: `${$DAYS[CreatedAt;$TODAY]} > 7`
$WEEK(param)	Returns the week-number of the given parameter, which can be `$TODAY` or a column.	

Function	What it does	Example
$MONTH(param)	Returns the month of the year as a number of the given parameter, which can be $TODAY or a column.	
$YEAR(param)	Returns the year as a number of the given parameter, which can be $TODAY or a column.	
$USERID	The ID of the currently logged in user.	
$USERNAME	The name of the currently logged in user.	
$PROJECTID	The ID of the currently selected project.	
$PROJECTNAME	The name of the currently selected project.	
$REPORTNAME	The name of the currently selected report.	
$REPORTID	The ID of the currently selected report.	

Below is the code of the pre-installed *Requirement with Child Requirements* report. With this report, a selected requirement is shown with its requirement ID. Full details regarding the requirement's child requirements are displayed. Although not a custom report, this report is a helpful example because it makes use of the `$PROJECTID` function. It also includes two parameters, reqID (requirement ID) and reqProp_Obsolete_0 (show obsolete requirements).

```
SELECT r.ReqID, r.ReqCreated, r.ReqName, r.TreeOrder
   FROM RTM_V_Requirements r INNER JOIN
   TM_ReqTreePaths rtp ON (rtp.ReqNodeID_pk_fk = r.ReqID)
   WHERE rtp.ParentNodeID_pk_fk=${reqID|22322|Requirement ID} AND
   r.ProjectID = ${$PROJECTID} AND
   r.MarkedAsObsolete=${reqProp_Obsolete_0|0|Show obsolete Requirements}
   ORDER BY r.TreeOrder ASC
```

Writing Advanced Queries with SQL

Advanced reports can be created through manual SQL coding. Virtually any reporting option is available if you know the database schema. Clicking **Advanced Query** hides the query string list boxes and opens the **Report data query** text box in which you can insert existing code or write new SQL code.

One approach is to begin query-string construction using the list boxes as outlined in *Creating New Reports*. If the report criteria are valid, the equivalent SQL statement will be generated and displayed, and then move to advanced mode for further modifications.

Note: You cannot move from advanced mode back to simple mode.

To write an advanced query directly in SQL:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the folder in which you want the new report to display.
   This determines where the report is stored in the directory tree.
3. Click on the toolbar. The **Create New Report** dialog box opens.
4. Type the name of the new report.
   This is the name that is displayed in the **Reports** tree.

5. Check the **Share this report with other users** check box if you want to make this report available to other users.
6. Type a description of the report in the **Description** text box.
7. Click **Advanced Query** to open the **Report data query** text box. Insert previously written code as necessary, or write new code directly in the text box.

   To assist you in writing SQL queries, a list box of Test Manager function placeholders is available. For details, see *SQL Functions for Custom Reports*. To insert one of the available pre-defined functions, select the corresponding placeholder from the **Insert placeholder** list box.

   Note: If you manually edit SQL code for the query, click **Check SQL** to confirm your work.
8. Once you have completed editing the report's properties, click **Finish** to save your settings.

Working with Sub-Reports

Adding Sub-Reports

To aggregate the results from multiple reports into the currently selected report, you can add sub-reports. When adding a report as a sub-report, the result columns and rows of the sub-report are concatenated to the results of the selected report.

To add a report as a sub-report:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Properties** tab.
4. Click **Add Sub-Report**.

   The **Add Sub-Report** dialog box displays.
5. From the **Reports** tree, select the sub-report you want to append to the current report.
6. Click **OK** to complete the addition of the sub-report. Sub-reports are displayed on the associated report's **Properties** page in the **Sub-Reports** section.

Deleting Sub-Reports

To delete a sub-report:

1. In the menu, click **Reports** > **Details View**.
2. Select the report in the **Reports** tree that has the associated sub-report that you want to delete.
3. Click the **Properties** tab.
4. Click ✕ in the **Actions** column of the sub-report that you want to delete.
5. Click **Yes** on the confirmation dialog box to confirm the deletion.

Report Templates

Uploading Report Templates

To upload a template from your local system:

1. In the menu, click **Reports** > **Details View**.
2. Select the report to which you want to associate the template.
3. Click the **Report** tab.
4. Click the **Click here to upload a new report template** link to open the **Upload Template** dialog box.
5. Give the template a meaningful **Name** and **Description**.

6. In the **Projects** list box, select the project to which you would like to make the template available or select **All Projects** to have the template associated with all projects.
7. Click **Browse** to browse to and select the template on your local system.
8. Click **OK** to upload the template.

Downloading Report Templates

Test Manager report templates render report data into formats that meet your specific needs. Templates can take the form of Excel spreadsheets, BIRT RCP Designer templates, XML, or CSV files. Downloading Test Manager report templates to your local system enables you to edit them through BIRT Report Designer or Microsoft Excel. After you download and edit a report, you can upload it to make it available to other users. For details see the related *Uploading Report Templates* procedure.

To download an existing report template for editing:

1. In the menu, click **Reports** > **Details View**.
2. Select a report that utilizes the template you want to modify from the **Reports** tree.
3. Click the **Properties** tab.
4. Click the download link of the template you want to download.

   The available download links are:

Link	Description
Download Excel Report Template	You receive an Excel file with a sheet named DATA that contains the data, for example in CSV format. This is the only affected sheet in the template, so you can specify information in adjoining sheets, for example diagrams.
Download BIRT Report Template	You receive the report data as an empty generic BIRT report template. The datasource is already configured.
Download as CSV	You receive the report data as a Comma Separated Values (CSV) file. Depending on your local settings, you will receive ‘,’ or ‘;’ as the delimiter character. The date is also formatted based on user settings.
Download as XML	You receive the report data as XML. The advantage of this approach over CSV is that you retain all subreport data. Accessing data outside of Test Manager - You can call a specific URL that offers the report data using the following format: `http://server/servicesExchange?hid=reportData&userName=<username>&passWord=<password>&reportFilterID=<ID of the report>&type=<csv|xml>`.

5. The **File Download** dialog box displays. Click **Save** and download the report file to your local system as a `.rptdesign` or `.xls` file, depending on the report type that you are downloading.
6. Edit the report based on your needs using either the BIRT RCP Designer, for `.rptdesign` files, or Excel, for `.xls` files.

Removing Report Templates

To remove the template of the current report:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the report from which you want to delete a template.
3. Click the **Report** tab.
4. Click ✕.
5. Click **Yes** on the subsequent confirmation dialog box.

Working with Charts

Displaying Charts

To display a chart:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree for which you want to view a chart.
3. Click the **Chart** tab to display the default chart.
4. To select a required chart type, click .
5. On the **Select Chart Type** dialog box, select a chart type from the list box.
6. Check the view properties that you want to apply to the chart.
   - 3D view
   - Show horizontal grid lines
   - Show vertical grid lines
   - Show legend
7. Specify how these chart options are to be saved:
   - Click the **For current user only** option button to have these chart settings override the report's standard settings whenever the current user views this chart.
   - Click the **As report standard** option button to have these chart settings presented to all users who don't have overriding user settings defined. This setting does not effect individual user settings.
8. Click **OK** to display the new chart type.

Note: The chart configurations you define here become the defaults for this report. When standard charts and graphs are not able to deliver the specific data that you require, or when they cannot display data in a required format, you can customize the appearance of queried data using the Test Manager reporting functionality. To open the current chart in a separate browser window, click at the top of the **Chart** page.

Printing Charts

To print the current chart:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Chart** tab.
4. Click **Print**. The chart data displays in a new window in printable format. Your system's print dialog box is also displayed.
5. Configure print settings as necessary and click **OK** to print the chart.

Removing Charts

Removing a chart only removes the currently selected chart template from the selected report, it does not remove the chart template entirely.

To remove the current chart template from the selected report:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Chart** tab.
4. Click **Remove chart type**. The **Remove Chart** dialog box opens.
5. Select one of the following:

- Select **Remove user settings (and revert to report standard)** to have the current user's chart settings deleted along with the chart. The chart will subsequently be displayed according to the report's standard settings. If no standard settings have been defined, the chart cannot be displayed.

  Note: This option is only available when the current user has defined specific chart settings.

- Select **Remove standard chart settings** of report to have any standard settings deleted along with the chart. User-specific settings are not affected by this option.

  Note: This option is only available when standard chart settings have been defined for a report.

6. Click **OK** to delete the chart template. If required, you can click the **<Click here to choose a chart type>** link to assign a new chart template to the selected report.

Report Chart Page

Reports > **Report View** > **Chart**

The **Chart** page enables you to define charts and graphs for data analysis. The page relies on the internal reporting engine of Test Manager to create standard charts and graphs from the data retrieved by the selected report query.

The following standard chart types are available:

- area chart
- bar chart
- horizontal stacked bar chart
- line chart
- pie chart
- stacked bar chart

Four display properties are also configurable for each chart type.

Customizing Reports with BIRT

Customizing BIRT Report Templates

With BIRT RCP Designer (BIRT), you can customize the pre-installed report templates of Test Manager and create custom report templates. For details on using BIRT, refer to the *SilkCentral Administration Module Help* and the BIRT RCP Designer documentation.

To download an existing template for editing:

1. In the menu, click **Reports** > **Details View**.
2. Select a report that utilizes the BIRT Report Template.
3. Click the **Properties** tab.
4. Click **Download BIRT report template**. You receive the report data as an empty generic BIRT report template. The datasource is already configured.
5. Once you have saved the template to your local system, modify it as required.

   For detailed information on configuring BIRT report templates, refer to the *SilkCentral Administration Module Help*.
6. Upload the modified report template by using the **Upload** link on **Administration** > **Reports** > **Report Templates**.

Reporting Area

Viewing Reports

Because each template expects a certain data format to produce a useful graph, not all templates can be applied to all report queries. You will receive an error message if you attempt to generate a report through an incompatible report template. For example, selecting the *Four Values Per Row As Horizontal Bar* template to display the *Requirements Status Overview* report works because this particular Microsoft Excel template requires exactly the four values, `failed`, `passed`, `not executed`, and `not covered` that the report query delivers.

To generate a report:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the report that you want to generate.
3. Click the **Report** tab.
4. Click the **Select Report Template** icon. The **Select Report Template** dialog box opens.
5. Select the template you wish to use.
6. Click **OK** to display the report.
7. *Optional:* If necessary, select an alternate view magnification for the report from the list box.
   100% is the default magnification. Other options are 50%, 75%, 150%, and 200%.

Saving Reports

How you save a report locally depends on whether you have selected a BIRT report template or an Excel template. If you have selected an Excel template, simply click the **Download Excel report template** link in the **Properties** page of the selected report. This will invoke Microsoft Excel on your local computer and the report will be loaded automatically.

If you have selected a BIRT report template, use the following procedure to save the report.

To export the current BIRT report as PDF:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the report that you want to save.
3. Click the **Report** tab.
4. Click on the **Report view** toolbar.
5. On the **File Download** dialog box, click **Save** to save the PDF document to a location of your choice.

Bookmarking Reports

The **BOOKMARK** button bookmarks the currently displayed report, including the parameters that you have set in the **Parameters** page. You can send bookmark URLs to other Test Manager users, allowing them to view reports with a single click.

The bookmark URL contains the parameters, prefixed with `rp_`. Date values are represented as the correlating `Long` values in UTC in the URL.

Viewing a Report as a PDF

To view the current report in PDF format within the report browser frame:

1. In the menu, click **Reports** > **Details View**.
2. In the **Reports** tree, select the report that you want to view.
3. Click the **Report** tab.
4. Click on the report view toolbar. The report displays in PDF format.

Accessing Most Recently Used (MRU) Reports

To select a recently-viewed report:

1. In the menu, click **Reports** > **Details View**.
2. On the **Reports** toolbar, expand the **Last Used Reports** list box.
3. Select the report that you want to view.

Reports Toolbar Functions

Test Manager > **Reports**

The **Reports** toolbar provides important commands for report management. The toolbar includes the following items:

Item	Icon	Description
New Child Folder		Enables creation of new report folders. Click **New Child Folder** to define a name and optional description for a new folder. The new folder displays as a child of the currently selected node in the **Reports** tree.
New Child Report		Enables creation of new reports. Click **New Child Report** to define a new report using the **Create New Report** dialog box. A new report displays as a child of the currently selected node in the **Reports** tree.
Edit		Edit a report.
Delete		Delete a report.
Cut		Cut a report and add it to the clipboard.
Copy		Copy a report to the clipboard.
Paste		Paste a report from the clipboard into the **Reports** tree.
Paste as Child		Paste a report from the clipboard into the **Reports** tree, as a subnode of the selected node.
Move Up		Move reports up within the **Reports** tree.
Move Down		Move reports down within the **Reports** tree.
Last Used Reports		Lists the Most Recently Used (MRU) reports by date/time in descending order. Select a report name from the list to advance to that report. Each time a report is accessed by clicking the **Data**, **Chart**, or **Report** tab, that report is added to the top of the list box. Accessing a report's **Properties** or **Parameters** tab does not result in that report being added to the **Last Used Reports** list box. The **Last Used Reports** list box is empty for new users and users who have not yet generated a report. The number of reports that displays in this list can be configured by your administrator. Refer to the *SilkCentral Administration Module Help* for details.

Report Properties

Editing Report Properties

To edit the properties of a report:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Report** tab.
4. Click the **Properties** tab.
5. Click **Edit**. The **Edit Report** dialog box displays.
6. Modify the **Name** and **Description** of the report as required.
7. Ensure that the **Share this report with other users** check box is checked if you intend to have this report shared with other users.
8. From the **Default tab** list box, select the tab that you want to be directed to when you select this report from one of the context-sensitive report lists.
9. Specify one of the following options to indicate how the report can be edited:

Option	Description
Simple report	You can modify the selection criteria—thus changing the results of the selected report—or you can click **Advanced Query** to modify the SQL query code.
Advanced report	If you have familiarity with SQL, you may edit the query code in the **Report data query** text box. To assist you in editing SQL queries, a list box of function placeholders (for example, “variables”) is available. To insert one of the available pre-defined functions, select the corresponding placeholder from the **Insert placeholder** list box.

- *Advanced report:* If you have familiarity with SQL, you may edit the query code in the **Report data query** field. To assist you in editing SQL queries, a list box of function placeholders, for example “variables”, is available. To insert one of the available pre-defined functions, select the corresponding placeholder from the **Insert placeholder** list box.

Note: If you manually edit the SQL code for the query, upon finishing, click **Check SQL** to confirm your work.

10. Click **Finish** to save your changes.

Report Properties Page

Reports > **Report View** > **Properties**

The **Properties** page displays the basic properties of the selected report, enabling you to edit these properties or the report templates. You can also add subreports to your reports.

The page displays the following items:

Items	Description
Report Name	Name of the report (customizable)
Report ID	System-defined identifier of the report
Description	A description of the report (customizable)
Created On	Date the report was created. Default reports are created when a database is created and connected to.

Items	Description
Created By	User who created the report. Default reports are created by the user Admin.
Changed On	Date the report was last modified.
Changed By	User who last modified the report.
Renderer	Report template that is currently assigned to the report.
Default Tab	Tab you are directed to when you select this report from one of the context-sensitive report lists.
Edit	Click to open the **Edit Report** dialog box.
Add Subreport	Click to add a subreport to the report.
Report Templates	The available pre-installed report templates are:

Link	Description
Download Excel Report Template	You receive an Excel file with a sheet named DATA that co the only affected sheet in the template, so you can specify i diagrams.
Download BIRT Report Template	You receive the report data as an empty generic BIRT repo
Download as CSV	You receive the report data as a Comma Separated Values will receive ‘,’ or ‘;’ as the delimiter character. The date is al
Download as XML	You receive the report data as XML. The advantage of this data. Accessing data outside of Test Manager - You can ca the following format: `http://server/servicesExch hid=reportData&userName=<username>&pass of the report>&type=<csv|xml>`.

Report Parameters

Editing Report Parameters

To edit the parameters of a report:

1. In the menu, click **Reports** > **Details View**.
2. Select a report in the **Reports** tree.
3. Click the **Parameters** tab. If the report has parameters defined for it, the parameters are listed here.
4. Click **Edit Parameters**. The **Edit Parameters** dialog box displays.
5. Edit the **Label** or **Value** of the listed parameters as required.
6. From the **Usage** list box, select the usage type of the parameter.
   - constant value
   - start time
   - end time
7. Click **OK** to save your changes.

Report Parameters Page

Reports > **Report View** > **Parameters**

The **Parameters** page lists customizable statement elements. Parameters can be defined any time before a report execution by simply changing them on the **Parameters** page. The syntax of a parameter is: `${parametername|defaultvalue|guiname}`. The `defaultvalue` and the `guiname` are optional. Parameter-names cannot contain whitespace characters.

When a report has parameters associated with it, it is possible to edit the values of the parameters before each report execution. Parameter values are stored in the current user context, which means edited values are available only to the user who performs the edits. When parameter values are not specified for a given report execution, the default values from the report definition are used.

Using the **Usage** list box, you can select the usage type of a parameter. The possible values are `constant value`, `start time`, and `end time`. `start time` and `end time` are used for reports that query for a specific date range.

When a report has subreports assigned to it, the parameters of those subreports are also shown in the **Parameters** page and the values are stored only within the context of the selected report. For example, the values are only used in conjunction with the current subreport configuration. When creating new reports, parameters are the values that are defined on the **Create New Report** dialog box in the **Selection criteria** area.

Report Data Page

Reports > **Report View** > **Data**

The **Data** page serves as a read-only result preview that shows the result parameters and values of the selected report in tabular format. The first row contains meta information about the current report that comes from the report template. Links included in the data are clickable and open the referenced location.

You can access the requirements, tests, or execution plans that you query for directly from the results list in the **Data** page. To do so, your query must include the column `ProjectID` and the respective ID of the element that you want to link to.

ID	Description
RequID	Query for this column to enable a link to requirements on the **Data** page of a report.
TestDefID	Query for this column to enable a link to tests on the **Data** page of a report.
ExecDefID	Query for this column to enable a link to execution plans on the **Data** page of a report.

If the query's result includes both the `ProjectID` and either of `RequID`, `TestDefID`, or `ExecDefID`, using exactly these terms as column names, the **Data** page will display the values in the element ID's column as a link. If you click such a link, Test Manager will switch to that element in the tree.

Check the **Show header** check box to display the additional meta properties for the report, like **Name**, **Project**, **Description**, and others, in tabular format in **Data view**.

Note: Only the first 100 result rows are displayed by default. You can select the following alternate view options from the list box.

- Show all rows
- Top 500 rows
- Top 1000 rows

Report Page

Reports > **Report View** > **Report**

The **Report** page is used to display data as a formatted report. If you have not yet assigned a template for your report, you can select one in the **Report** page.

A list box provides a selection of all available report templates. In addition to many system-installed templates, any custom report templates that were uploaded from **Administration** > **Reports** > **Report Templates** are also available here. For details on setting up and uploading custom report templates, refer to the *SilkCentral Administration Module Help*.

You can also download an existing template by clicking the **Properties** tab, and then clicking the download link that corresponds to the report format you are working with, Excel, BIRT, CSV, or XML. From there you can customize the template to your needs.

Note: Reports are cached to improve the performance of reporting. Click **Update** to update the report data immediately.

Project Overview Report

Reports > **Project Overview Report**

Displays the **Project Overview Report**, which offers a high-level overview of the status of the selected project. The **Project Overview Report** includes the following sections:

Section	Description
General Report Information	General information like the name of the current project, the report description, and the planned release date.
Requirements	Test coverage status for requirements. Shows the percentage of test coverage in tabular and graph format both all requirements and requirements with high priority.
Tests	Test-type distribution and test-execution results in chart and tabular form for tests.
Issues	The amount of found, fixed, deferred, and currently open in the backlog issues, that are linked to tests, in tabular and graph format.

Code Analysis

The goal of conventional code coverage is to deliver information about what code is covered by tests. This approach is typically used to gain code-coverage information for unit tests. The code-analysis functionality in Test Manager goes well beyond this, delivering data for unit, functional, and load tests, both automated and manual, in managed environments. Code coverage measurements are utilized to track test progress and guide test planning. This innovative approach in Test Manager to code coverage draws on the relationship between specific tests and the code they test. This approach enables you to perform impact/dependency analysis of code changes from the testing perspective. It also assists you in optimizing your testing, by helping you identify the test runs that are most relevant to a specific code change.

The **Code Analysis** unit offers code-coverage data for AUT (Applications Under Test), packages/namespaces, classes, methods, and statements, enabling you to perform test-impact analysis, which determines the tests that should be run in response to specific code changes, and effort analysis, which determines how many hours of automated testing and manual testing are required to adequately cover specific code changes. Each of these code-analysis tasks can be addressed by running pre-configured reports. The unit features a navigation tree that lists all products that have been created for the selected project. You can drill down into products to select specific versions, and at the deepest level, specific builds.

Enabling Code Analysis

You must configure Test Manager to gather code coverage data from an application under test. You can configure any number of execution plans listed in **Executions** > **Details View** > **Deployment**.

Java Code Analysis

Java Limitations and Prerequisites

The following sections display the known limitations and prerequisites that must be performed to enable java code analysis with Test Manager.

White Spaces Are Prohibited

Whitespaces cannot be ignored or added when modifying batch files or syntax. Otherwise, `java.lang.error` messages will be encountered. For example: `Please note that unnecessary whitespace within the command line will result in an error on initiation as follows. Exception in thread "main" java.lang.NoClassDefFoundError: ….. Directory not empty`.

You Must Start the Application Under Test Before You StartTest Manager

The application under test must be started before code coverage execution is kicked off. The AUT must be started independently of Test Manager so that code coverage can hook into it.

Java Prerequisites

To find out which classes and methods of an application under test (AUT) have been invoked within jar files, you must locate the file `sctmcc.dll` in the directory from which the AUT is executed. You can download this file directly from your Test Manager GUI by selecting **Help** > **Tools** > **Java Code Analysis Agent** . Code analysis works with Java versions 1.4, 1.5, and 1.6. However, there is a difference in how the DLL can be loaded when starting up the virtual machine (VM).

When starting up the application under test from the command line, the following arguments must be passed to the VM in order to load sctmcc.dll:

Java 1.4	`-Xrunsctmcc:port=19129,<options>`
Java 1.5 and 1.6	`-agentlib:sctmcc=<options>`

Java Code Analysis Options

You can use the following options for all supported Java versions:

Option	Description
`port=19129`	Port of code coverage service.
`coveragetype="line"`	Possible values are `line` or `method`. To get code coverage information with Test Manager, you must use `line`.
`coveragepath={"library1.jar"; "library2.jar"}`	Jar-libraries to monitor for code coverage information.
`name="ServerName"`	Name of the monitored application.
`filters={profilingExample.*,java.lang.*,!java.util.*}`	For large JAR files, you can create filters to define for what sub-set of the classes included in the JAR coverage data is generated. The

Option	Description
	filters in the above example specify that coverage data is generated for the classes included in the `profilingExample` and `java.lang` packages and sub-packages. The exclamation mark defines that no coverage data is generated for the classes included in the `java.util` package and all sub-packages. **Note:** When both inclusions and exclusions are specified in the filters, the exclusions are evaluated first. If a class is both included and excluded through filters, this class will be excluded and no coverage data is generated for this class. If the filter option is not specified, coverage data is generated for all used classes, unless the coverage path option is set, in which case coverage data is generated for all classes in the coverage path.

Java Code Analysis Examples

Java 1.5 and 1.6:

```
"<java_home_directory>\bin\java" -agentlib:sctmcc=port=19129,
  coveragetype="line",
  coveragepath={"C:\dev\deploy\lib\library1.jar";"C:\dev\deploy
\lib\library2.jar"},
  name="myAUTserver" com.caucho.server.http.HttpServer,
  filters={java.lang.*,!java.util.*}
```

Java 1.4:

```
C:\Java\j2sdk1.4.2_06\bin\java.exe -Xrunsctmcc:port=19129,
  coveragetype="line",
  coveragepath={"C:\Program Files\Silk\SilkTest\JavaEx\JFC
\swingall.jar";
  "C:\Program Files\Micro Focus\SilkTest\JavaEx\JFC
\Swing11TestApp.jar"},
  name="Test Application",
  filters={java.lang.*,!java.util.*} -
Dsilktest.tafont=arialuni.ttf -cp .;%FontDir% ta
```

.NET Code Analysis

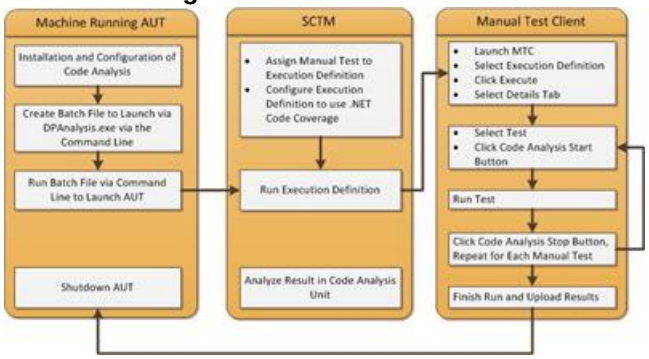
.NET Code Analysis and Manual Testing

The following sections detail the required steps to install and use the DevPartner .NET code coverage components shipped with Test Manager. The example below will get you started by using manual testing and a single AUT. If you plan to use automated testing, first read and understand the steps below, and then see the topic titled *.NET Code Analysis and Automated Testing* .

Note: In order to use the Test Manager/.NET code coverage functionality, the Test Manager - DevPartner components must be used. The Test Manager-DevPartner components cannot be installed on machines that have DevPartner 10.1 or below previously installed.

Note: .NET code analysis requires the PDB files generated at compile time of the .NET application under test (AUT) to be located in the same directory as the AUT's executable(s). If the PDBs for the AUT are not available, it is not possible to leverage the .NET code analysis functionality.

Manual Testing Workflow


Machine Running AUT
Installation and Configuration of Code Analysis
Create Batch File to Launch via DPAnalysis.exe via the Command Line
Run Batch File via Command Line to Launch AUT
Shutdown AUT
SCTM
Assign Manual Test to Execution Definition
Configure Execution Definition to use .NET Code Coverage
Run Execution Definition
Analyze Result in Code Analysis Unit
Manual Test Client
Launch MTC
Select Execution Definition
Click Execute
Select Details Tab
Select Test
Click Code Analysis Start Button
Run Test
Click Code Analysis Stop Button, Repeat for Each Manual Test
Finish Run and Upload Results

Installing the Windows Code Analysis Framework and DevPartner Analytics

1. Go to the machine that hosts the application to test.
2. Open Test Manager.
3. Go to **Help** > **Tools** and click `Windows Code Analysis Framework`. This will prompt a download of `CAFramework.exe`. `CAFramework.exe` includes the installation for the `Code Analysis Framework` and the `DevPartner Analytics` package.

   Note: Multiple execution servers running tests against the same Code Analysis Framework at the same time is not supported.
4. Double click the download file to start the installation. Walk-through the installation of the `Windows Code Analysis Framework`. At the end of the installation, you will be prompted to install `DevPartner Analytics`. Ensure that the **DevPartner Analytics Installation** checkbox is checked on the appropriate installation screen and install this component. At the end of this installation, all required files are installed and the proper service is started. The name of the service is `SilkCentral Code Analysis Service`.

Configuring DPAnalysis to launch Your AUT

1. Create a `.BAT` file that points to `DPAnalysis.exe`. An example would be:

```
"C:\Program Files\Micro Focus\DevPartner\Analysis\dpanalysis.exe" /cov /
USE_AUTOMATION_INTERFACE
/W "C:\Common\Development\MYAPP"
/P "C:\Common\Development\MYAPP\MyApp.exe"
```

2. For readability, the example above is on multiple lines. Your `.BAT` file should be on one line. Make the following modifications:
   a) Ensure that the first line is your installation path to `dpanalysis.exe`. The two switches are mandatory.
   b) The second line contains a mandatory switch and needs to include the path to your executable and PDBs.
   c) The third line contains a mandatory switch and needs to include the absolute path to your test application and the application name.
3. Double-click the `.BAT` file to have DPAnalysis launch your application.

For more information on running `DPAnalysis.exe`, see the topic titled *Using DpAnalysis.exe*. That topic and related topics contain information on additional switches and how to use a configuration file if you have multiple applications to test.

Configuring Test Manager for Code Analysis

1. In the menu, click **Execution Planning** > **Details View**.
2. Open an existing execution plan that contains a manual test with steps to test functionality for your AUT.
3. Click the **Deployment** tab.
4. Under **Code Analysis Settings**, click **Edit** to view the **Edit Code Analysis Settings** dialog box.
5. Ensure that **Enable Code Analysis** is checked.
6. Select `DevPartner .NET Code Analysis` from the **Code Analysis Profile** list.
7. For the `Hostnames` field, type in the host name followed by colon and the port number. For example: `localhost:19129`.

   The default code analysis framework port is `19129`. You can configure the code analysis framework to run at a given port using the following xml file: `<install dir>\Code Analysis \CodeAnalysisConfig.xml`
8. Click **OK**.
9. Run the test by clicking **Run** in the **Execution** area.
10. On the **Run** dialog box, ensure that **All Tests** is checked.
11. In the **Set build for execution plan**, select the proper build for your execution plan.
12. Click **OK**.

Running Manual Tests

1. Start the **Manual Testing Client**.
2. Double click your manual test that appears in the **Inbox** of the **Manual Testing Client**.
3. On the **Execute Test** dialog box, click the **Details** tab.
4. Click **Start** situated next to the **Code Analysis** label.
5. On your AUT, perform your manual tests.
6. When you have finished, return to the **Manual Testing Client** and click **Stop**.

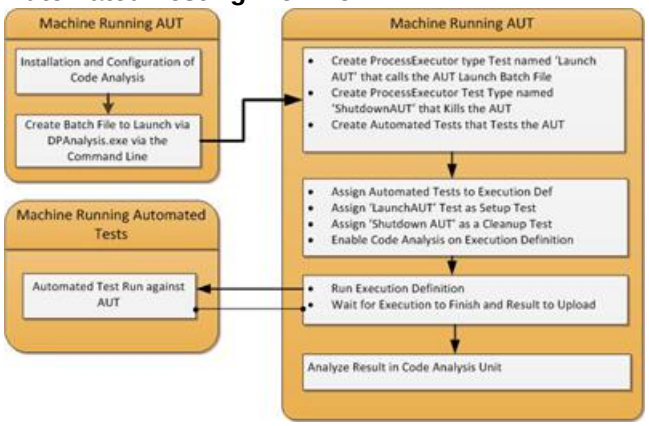
   Do not use **Resume** or all previous testing coverage will be lost and only the analysis for the last **Resume-Stop** will be reported.
7. In the **Test Steps** list, give each step a proper status.
8. Click **Finish Run** and then **Close**.
9. Click **Upload** on the **Manual Testing Client**.
10. Return to Test Manager and click **Reports** > **Code Analysis**.
11. Click the proper **Product** and **Build** nodes and then review your code coverage results.

.NET Code Analysis and Automated Testing

This section describes two scenarios that you may encounter when executing your tests that use code coverage. It also provides proper ways to structure your tests within an execution plan.

The examples used in the following topics are specific to DevPartner Code Analysis, but the structure is similar for other code analysis tools. For example, you will have different startup/cleanup parameters in your execution plan.

Automated Testing Workflow


Machine Running AUT
Installation and Configuration of Code Analysis
Create Batch File to Launch via DPAnalysis.exe via the Command Line
Machine Running Automated Tests
Automated Test Run against AUT
Machine Running AUT
Create ProcessExecutor type Test named 'Launch AUT' that calls the AUT Launch Batch File
Create ProcessExecutor Test Type named 'ShutdownAUT' that Kills the AUT
Create Automated Tests that Tests the AUT
Assign Automated Tests to Execution Def
Assign 'LaunchAUT' Test as Setup Test
Assign 'Shutdown AUT' as a Cleanup Test
Enable Code Analysis on Execution Definition
Run Execution Definition
Wait for Execution to Finish and Result to Upload
Analyze Result in Code Analysis Unit

Configuring One or More Tests with One AUT

If you have the scenario where you have one or more tests in a single execution plan and they test against a single application under test, you should structure your testing as follows:

1. Create a `ProcessExecutor` test that will start your code analysis application.

   The `ProcessExecutor` **Argument List** should start your code analysis program. For example: `start cmd /c start cmd /c D:\sctm\dps\startDPS_Simple1.bat`, where `startDPS_Simple1.bat` contains:

   ```
   "C:\Program Files\Micro Focus\DevPartner\Analysis\dpanalysis.exe"
   /cov
   /USE_AUTOMATION_INTERFACE
   /O "D:\SCTM\temp"
   /W "D:\SCTM\DPS\TESTAPP\SCTMCodeAnalysisTestApp\bin\Debug"
   /P "D:\SCTM\DPS\TESTAPP\SCTMCodeAnalysisTestApp\bin\Debug
   \SCTMCodeAnalysisTestApp.exe"
   ```
2. Create an execution plan.
3. Click **Executions** > **Details View**, select an execution plan and click the **Setup/Cleanup** tab.
4. Add the previously created test as the **Setup Test**.
5. Add a test to the execution plan that automates your application under test with an automated testing tool.
6. *Optional:* Add another test to the execution plan that automates your application under test with an automated testing tool.
7. Create a `ProcessExecutor` test.
8. Click **Executions** > **Details View**, select an execution plan and click the **Setup/Cleanup** tab.
9. Add the previously created test as the **Cleanup Test**.

The `ProcessExecutor` **Argument List** should close your program: `start cmd /c taskkill /IM SCTMCodeAnalysisTestApp.exe`.

Configuring One or More Tests with Multiple AUTs

If you have the scenario where you have one or more tests in a single execution plan and they test against multiple AUTs, you should structure your testing as follows:

1. Create a `ProcessExecutor` test that will start your code analysis application.

   The `ProcessExecutor` **Argument List** should start your code analysis program. For example: `start cmd /c start cmd /c D:\sctm\dps\two_app_dps.cmd`, where `two_app_dps.cmd` contains:

   ```
   "C:\Program Files\Micro Focus\DevPartner\Analysis\dpanalysis.exe"
   /config D:\SCTM\DPS\configuration_file.xml
   ```

and `configuration_file.xml` contains:

```
<?xml version="1.0" ?>
<ProductConfiguration>
    <RuntimeAnalysis Type="Coverage" MaximumSessionDuration="1000"
NoUIMsg="true" />
    <Targets RunInParallel="true">
        <Process CollectData="true" Spawn="true" NoWaitForCompletion="true"
NM_USE_AUTOMATION_INTERFACE="1" >
            <AnalysisOptions NM_USE_AUTOMATION_INTERFACE="1" NO_MACH5="1"
NM_METHOD_GRANULARITY="1" SESSION_DIR="c:\temp" />
            <Path>D:\SCTM\DPS\SCTMCodeAnalysisTestApp
\SCTMCodeAnalysisTestApp\
          SCTMCodeAnalysisTestApp\bin\Debug\SCTMCodeAnalysisTestApp.exe</
Path>
            <Arguments></Arguments>
            <WorkingDirectory>D:\SCTM\DPS\SCTMCodeAnalysisTestApp
\SCTMCodeAnalysisTestApp\
        SCTMCodeAnalysisTestApp\bin\Debug</WorkingDirectory>
        </Process>
        <Process CollectData="true" Spawn="true" NoWaitForCompletion="true"
>
            <AnalysisOptions NM_USE_AUTOMATION_INTERFACE="1" NO_MACH5="1"
NM_METHOD_GRANULARITY="1" SESSION_DIR="c:\temp" />
            <Path>D:\SourceCode\SVNQAD\Development\Development\SOURCECODE\
         QADSCTMMigration\bin\QADSCTMMigration.exe</Path>
            <Arguments></Arguments>
            <WorkingDirectory>D:\SourceCode\SVNQAD\Development\Development\
         SOURCECODE\QADSCTMMigration\bin</WorkingDirectory>
        </Process>
    </Targets>
</ProductConfiguration>
```

2. Create an execution plan.
3. Click **Executions** > **Details View**, select an execution plan and click the **Setup/Cleanup** tab.
4. Add the previously created test as the **Setup Test**.
5. Add a test to the execution plan that automates your first application under test with an automated testing tool.
6. Add another test to the execution plan that automates your second application under test with an automated testing tool.
7. Create a `ProcessExecutor` test.
8. Click **Executions** > **Details View**, select an execution plan and click the **Setup/Cleanup** tab.
9. Add the previously created test as the **Cleanup Test**.

The `ProcessExecutor` **Argument List** should close your programs: `start cmd /c taskkill /IM application1.exe /IM Application2.exe`.

Using DPAnalysis.exe

`DPAnalysis.exe` is the DevPartner code coverage application that allows you to profile .NET applications. The application is started using the command line.

You can use the application with traditional switches as documented in these topics, or you can add an additional switch that points to a configuration file. You may want to use a configuration file if you have an exceptionally long command line string or you want to run multiple applications in a single session.

The `/USE_AUTOMATION_INTERFACE` switch is mandatory. It causes the automation interface environment needed for the interaction with the framework to be established. It also causes license checking to be bypassed. This switch does not take any parameters.

Running DPAnalysis.exe from the Command Line

The DevPartner Studio Code Coverage installation includes `DPAnalysis.exe`, a command line executable that is installed in the `\Program Files\Micro Focus\DevPartner Studio\Analysis\` directory.

Use the following syntax and switches to run the DevPartner Studio Code Coverage tool from the command line:

```
DPAnalysis  [/Cov] [/USE_AUTOMATION_INTERFACE] [/E|/D|/R]
[/W workingdirectory] [/PROJ_DIR] [/H hostmachine]
[/NOWAIT] [/NO_UI_MSG] [/N] [/F] [/A C:\temp1;C:\temp2]
[/NO_QUANTUM /NM_METHOD_GRANULARITY /EXCLUDE_SYSTEM_DLLS
/NM_ALLOW_INLINING /NO_OLEHOOKS /NM_TRACK_SYSTEM_OBJECTS]
{/P|/S} target [target arguments]
```

Example

You can use `DPAnalysis.exe` directly from the command line, using switches to direct the analysis session. For example, the following command line launches a code coverage session for the application *SCTMCodeAnalysisTestApp.exe*.

```
dpanalysis.exe /cov /USE_AUTOMATION_INTERFACE /W "D:\SCTM
\DPSCoverageIntegration\Test App\
SCTMCodeAnalysisTestApp\SCTMCodeAnalysisTestApp\bin\Debug" /P
"D:\SCTM\DPSCoverageIntegration\Test App\SCTMCodeAnalysisTestApp
\SCTMCodeAnalysisTestApp\bin
\Debug\SCTMCodeAnalysisTestApp.exe"
```

Switches

Analysis Type Switches

Name	Description
`/Cov[erage]`	Sets analysis type to DevPartner Coverage Analysis

Note: `DPAnalysis.exe` does not instrument unmanaged code. To collect performance or coverage analysis data for an unmanaged application, you must first instrument the application.

Data Collection Switches

Enables or disables data collection for a given target, but does not launch the target. Optional.

Name	Description
`/E[nable]`	Enables data collection for the specified process or service.
`/D[isable]`	Disables data collection for the specified process or service.
`/R[epeat]`	Profiling will occur any time you run the specified process until you use the /D switch to disable profiling.

Other Switches

These switches are optional.

Name	Description
`/W[orkingDir]`	Specify the working directory for the target process or service.
`/PROJ_DIR`	Specify directory of DevPartner Studio project, used to locate playlists, etc.

Name	Description
/N[ewconsole]	Run the process in its own command window. If you use DPAnalysis.exe to analyze a console application that requires keyboard input, you must use the /NewConsole switch to open a console window to accept the input.
/INCLUSION_FILTER <filterFile>	The /INCLUSION_FILTER switch is optional and allows an Inclusion Filter file to be specified. This file is used when the coverage data is provided to the framework in the SCTM-specific XML format. It causes the report generation program to filter out all classes that are not specified in the filter file. This switch takes a single parameter as shown here. /INCLUSION_FILTER <filterFile> where <filterFile> is the fully-qualified path of the filter file to be used for the duration of the program. The sample Inclusion_Filter file below is a simple XML file that contains the set of classes to be included in the generated report. It also contains a value, <filterClasses>, which indicates whether the filter is to be applied. <?xml version="1.0" encoding="utf-8"?> <!--D:\Public\SCTM Work\ProdAPI_Tester_CSharp_LINES.dpcov--> <inclusionInfo> <filterClasses>false</filterClasses> <includedClass>MicroFocus.Products.Tools.TCS_TraceMechanism.CTCS_MsgStream</includedClass> <includedClass>MicroFocus.Products.Tools.TCS_TraceMechanism.CTCS_RecvTask</includedClass> <includedClass>MicroFocus.Products.Tools.TCS_TraceMechanism.CTCS_TimeStamp</includedClass> <includedClass>MicroFocus.Products.Tools.TCS_TraceMechanism.CTCS_Tracer</includedClass> <includedClass>MicroFocus.Products.Tools.TCS_TraceMechanism.CMsgRegisterClientReqMB</includedClass> </inclusionInfo>
/A[ssembly_dir]	For coverage analysis only. A list of paths that contain referenced DLL/PDB files. Use this switch to specify any referenced but not loaded DLL/PDB files to display in a coverage session file. Separate multiple paths with a semicolon (;).

Analysis Options

Example of the analysis options is:

```
<AnalysisOptions NM_USE_AUTOMATION_INTERFACE="1" NO_MACH5="1"
NM_METHOD_GRANULARITY="1" SESSION_DIR="c:\temp" />
```

Column	Type
/NM_USE_AUTOMATION_INTERFACE	A required switch in the analysis options section of the coniguration file. The value must equal "1".
/NO_QUANTUM	Disables excluding time spent on other threads.
/NM_ALLOW_INLINING	Enable run-time instrumentation of inline methods (Coverage and Performance only).

Column	Type
/NO_OLEHOOKS	Disable collection of COM.

Target Switch

Identifies target to follow as either a process or service. Required. Pick only one. All arguments that follow the target name or path will be arguments to the target.

Name	Description
/P[rocess]	Specify a target process (followed by arguments to process)
/S[ervice]	Specify a target service (followed by arguments to service)
/C[onfig]	Path to configuration file

DPAnalysis Configuration File

To manage analysis sessions with an XML configuration file, run `DPAnalysis.exe` from the command line with the `/config` switch and a properly structured XML configuration file as its target. For example:

```
dpanalysis.exe /config c:\temp\configuration_file.xml
```

By using a configuration file, you can profile and manage multiple processes or services. The ability to profile multiple processes can be especially useful for analyzing Web applications.

Sample Configuration File

```
<?xml version="1.0" ?>
<ProductConfiguration>
  <RuntimeAnalysis Type="Coverage" MaximumSessionDuration="1000"
NoUIMsg="true" />
     <Targets RunInParallel="true">
          <Process CollectData="true" Spawn="true" NoWaitForCompletion="true"
NM_USE_AUTOMATION_INTERFACE="1" >
               <AnalysisOptions NM_USE_AUTOMATION_INTERFACE="1" NO_MACH5="1"
NM_METHOD_GRANULARITY="1" SESSION_DIR="c:\temp" />
               <Path>D:\SCTM\DPS\SCTMCodeAnalysisTestApp
\SCTMCodeAnalysisTestApp\
        SCTMCodeAnalysisTestApp\bin\Debug\SCTMCodeAnalysisTestApp.exe</Path>
               <Arguments></Arguments>
               <WorkingDirectory>D:\SCTM\DPS\SCTMCodeAnalysisTestApp
\SCTMCodeAnalysisTestApp\
        SCTMCodeAnalysisTestApp\bin\Debug</WorkingDirectory>
          </Process>
          <Process CollectData="true" Spawn="true" NoWaitForCompletion="true"
>
               <AnalysisOptions NM_USE_AUTOMATION_INTERFACE="1" NO_MACH5="1"
NM_METHOD_GRANULARITY="1" SESSION_DIR="c:\temp" />
               <Path>D:\SourceCode\SVNQAD\Development\Development\SOURCECODE
\QADSCTMMigration\bin\QADSCTMMigration.exe</Path>
               <Arguments></Arguments>
               <WorkingDirectory>D:\SourceCode\SVNQAD\Development\Development
\SOURCECODE\QADSCTMMigration\bin</WorkingDirectory>
          </Process>
     </Targets>
</ProductConfiguration>
```

AnalysisOptions Element

Attributes that work with `AnalysisOptions` vary depending on the type of analysis session you run.

Note: DPAnalysis.exe ignores attributes mismatched with the type of analysis.

```
<AnalysisOptions
    SESSION_DIR = "c:\MyDir"
    SESSION_FILENAME = "myfile.dpcov"
    NM_METHOD_GRANULARITY = "1"
    EXCLUDE_SYSTEM_DLLS = "1"
    NM_ALLOW_INLINING = "1"
    NO_OLEHOOKS = "1"
    NM_TRACK_SYSTEM_OBJECTS = "1"
 NO_QUANTUM="1"
    NO_MACH5 = "1"
    FORCE_PROFILING = "1"
 ASSEMBLY_DIR = "C:\TEMP1;C:\TEMP2"
/>
```

SESSION_DIR

Optional. Use with Coverage, Memory, Performance, and Performance Expert. Specify a directory for saving the session file generated by the profiled target. Without this attribute, the resulting session file will be placed in the user's `My Documents` directory. If both `SESSION_DIR` and `SESSION_FILENAME` are absent, `DPAnalysis.exe` prompts you for the save location at the end of the session.

SESSION_FILENAME

Optional. Use with Coverage, Memory, Performance, and Performance Expert. Specify a session name for the session file generated for this target. Without this attribute, `DPAnalysis.exe` creates a unique name by combining the target's image name with a number (for example, `iexplore1.dpprf`). If you specify a name but no directory, the file will be placed in user's `My Documents` directory. If both `SESSION_FILENAME` and `SESSION_DIR` are absent, `DPAnalysis.exe` prompts you for the save location at the end of the session.

NM_METHOD_GRANULARITY

Optional. Use with Performance to set data collection granularity to method-level (line-level is default). Specify a value of `1` to set the attribute. Omitting the attribute disables it.

EXCLUDE_SYSTEM_DLLS

Optional. Use with Performance to exclude system images. Specify a value of `1` to set the attribute. Omitting the attribute disables it.

NM_ALLOW_INLINING

Optional. Use with Coverage and Performance to specify level of analysis detail. Enables run-time instrumentation of inline methods. Equivalent to the **Instrument Inline Functions** property. Specify a value of `1` to instrument inline functions. Omit the attribute to disable it.

NO_OLEHOOKS

Optional. Use with Performance analysis to activate tracking of system objects. Specify a value of `1` to set the attribute. Omitting the attribute disables it.

NM_TRACK_SYSTEM_OBJECTS

Optional. Use with Memory analysis to ignore system or third-party object allocations when tracking allocated objects. Specify a value of `1` to set the attribute. Omitting the attribute disables it. The default state (disabled) enables you to see memory allocations made when your application uses system or other non-profiled resources.

NO_QUANTUM

Optional. Use with Performance and Performance Expert analysis to exclude time spent in threads of other running applications. Specify a value of `1` to set the attribute. Omitting the attribute disables it.

FORCE_PROFILING

Optional. Use with Performance and Coverage analysis to force profiling of applications written without managed code or DevPartner Native C/C++ Instrumentation. Specify a value of `1` to set the attribute. Omitting the attribute disables it.

ASSEMBLY_DIR

Optional. Use with coverage analysis to specify the absolute path(s) containing referenced but not loaded assemblies to be shown in the Coverage session file. Separate multiple locations with a semicolon. Omitting the attribute disables it.

Element Information

Number of Occurrences	`0` or `1` per `Process` or `Service`.
Parent Elements	`Process`, `Service`
Contents	None

Remarks

Optional. Defines run time attributes for the specified target process or service. Attributes correspond to DevPartner Coverage, Memory, and Performance properties accessible from the Properties Window in Visual Studio.

Example

The following example shows a construction using AnalysisOptions within a Service.

```
<Service CollectData="true">
  <AnalysisOptions NM_METHOD_GRANULARITY="1" EXCLUDE_SYSTEM_DLLS="1"
NM_ALLOW_INLINING="1"
  NO_OLEHOOKS="1" ASSEMBLY_DIR="C:\TEMP">
</Service>
```

Arguments Element

Syntax

```
<Arguments>/arg1 /arg2 /arg3</Arguments>
```

Attributes

None

Element Information

Number of Occurrences	`0` or `1` per `Process` or `Service`.
Parent Elements	`Process`, `Service`
Contents	None

Remarks

Optional. No default if omitted. Arguments to be passed to the target process or service.

Example

The following example uses `Arguments` within a `Process` element.

```
<Process CollectData="true">
 <Arguments>/arg1 /agr2 /arg3</Arguments>
</Process>
```

ExcludeImages Element

Syntax

```
<ExcludeImages>
  <Image>ClassLibrary1.dll</Image>
  <Image>ClassLibrary2.dll</Image>
</ExcludeImages>
```

Attribues

None.

Element Information

Number of Occurrences	`0` or `1` per `Process` or `Service`.
Parent Elements	`Process`, `Service`
Contents	`Image`

Remarks

Optional. No default if omitted. Provide a list of at least one image (no maximum) which, if loaded by the target process or service, will not be profiled.

Example

The following example shows a construction using ExcludeImages within a Process element. Note the Image elements contained within ExcludeImages.

```
<Process CollectData="true">
 <ExcludeImages>
   <Image>ClassLibrary1.dll</Image>
   <Image>ClassLibrary2.dll</Image>
 </ExcludeImages>
</Process>
```

Host Element

Syntax

```
<Host>hostmachine</Host>
```

Attributes

None.

Element Information

Number of Occurrences	`0` or `1` per `Process` or `Service`.
Parent Elements	`Process`, `Service`.

Contents	Name of the host machine.

Remarks

Optional. No default if omitted. Sets the host machine of the target process or service.

Example

The following example shows a construction using `Host` within a `Service`. Note that the example includes the required `Name` element.

```
<Service CollectData="true">
  <Name>ServiceApp</Name>
  <Host>remotemachine</Host>
</Service>
```

Name Element

Syntax

```
<Name>MyServiceName</Name>
```

Attributes

None.

Element Information

Number of Occurrences	1.
Parent Elements	`Service`.
Contents	Service name.

Remarks

Required. The name of the service as registered with the service control manager. This is the same name you would use for the system's `NET START` command.

Example

The following example shows a construction using `Name` within a `Service`.

```
<Service CollectData="true">
 <Name>ServiceApp</Name>
</Service>
```

Path Element

Syntax

```
<Path> c:\MyDir\target.exe </Path>
```

Attributes

None.

Element Information

Number of Occurrences	1.

Parent Elements	`Process`.
Contents	Path to the executable.

Remarks

Required. Specify a fully qualified or relative path to the executable. You can specify the executable name without the path if the executable exists in the current directory.

Example

The following example shows a construction using Path within a Process element.

```
<Process CollectData="true">
  <Path>ClientApp.exe</Path>
</Process>
```

Process Element

Syntax

```
<Process CollectData="true or false"
   Spawn="true or false"
      NoWaitForCompletion="true or false"
      NewConsole="true or false"
   RepeatInjection="true or false"> ...
</Process>
```

Attributes

Profiling will occur any time you run the specified process until you use the /D switch to disable profiling.

CollectData	Optional. Specify **true** or **false**. Defaults to **true** if omitted. Specifies whether profiling will be enabled for the target process.
Spawn	Optional. Specify `true` or `false`. Defaults to `true` if omitted. Specifies if `DPAnalysis.exe` will spawn the specified target. Do not set to `true` for `aspnet_wp.exe` or `w3wp.exe`. DevPartner cannot spawn the `ASP.NET` worker process directly. Launch the `ASP.NET` worker process by opening the target Web page.
NoWaitForCompletion	Optional. Specify `true` or `false`. Defaults to **false** if omitted. The default is to wait until the process has completed. If set to `true`, causes `DPAnalysis.exe` to wait only until the target has started executing. `DPAnalysis.exe` will not wait for processes on remote machines (using the `Host` element). The `MaximumSessionDuration` attribute in the `RuntimeAnalysis` element overrides `NoWaitForCompletion`.
NewConsole	Optional. Specify `true` or `false`. Defaults to `false` if omitted. Causes `DPAnalysis.exe` to run the target in its own console window. The default is to use the same console that you typed the `DPAnalysis.exe` command line in. If you use `DPAnalysis.exe` to analyze a console application that requires keyboard input, you must use the `/NewConsole` switch to open a console window to accept the input.
RepeatInjection	Optional. Specify `true` or `false`. Defaults to `false` if omitted. Causes `DPAnalysis.exe` to profile the target in every time it runs until you explicitly specify `false`.

Element Information

Number of Occurrences	1 or more.
Parent Elements	`Target`.
Contents	AnalysisOptions, Path, Arguments, WorkingDirectory, ExcludeImages.

Remarks

Specifies a target executable.

Example

The following example shows a construction using Process and includes AnalysisOptions, Path, Arguments, and WorkingDirectory tags.

```
<Targets RunInParallel="true">
  <Process CollectData="true" Spawn="true" NoWaitForCompletion="true"
NewConsole="true">
    <AnalysisOptions NO_MACH5="1" NM_METHOD_GRANULARITY="1" SESSION_DIR="c:
\MyDir" />
    <Path>ClientApp.exe</Path>
    <Arguments>/arg1 /agr2 /arg3</Arguments>
    <WorkingDirectory>c:\temp</WorkingDirectory>
  </Process>
</Targets>
```

RuntimeAnalysis Element

Syntax

```
<RuntimeAnalysis Type="Coverage" NoUIMsg="false"
NM_INCLUSION_FILTER="Q:\AppDir\InclusionFilter.xml">
```

Attributes

Type	*Required:* Possible choices are: `Performance`; `Coverage`; `Memory`; or `Expert`. Specifies the analysis types for all targets listed.
MaximumSessionDuration	*Optional:* If omitted, no default used. If specified, `DPAnalysis.exe` will limit a session run for this amount of time. For example, if you specify `MaximiumSessionDuration="60"` and then begin profiling a service (with `RestartAtEndOfRun="true"` for the service), after `60` seconds, `DPAnalysis.exe` will stop the service and then restart the service.
NoUIMsg	*Optional:* If omitted, `false` is used by default. If set to `true` `DPAnalysis.exe` suppresses all UI error messages that may appear during the duration of the session. Setting this to `true` is useful when sessions are run unattended or when running a large number of consecutive tests.
NM_INCLUSION_FILTER	*Optional:* Allows an Inclusion Filter file to be specified. This file is used when the coverage data is provided to the framework in the SCTM-specific XML format. It causes the report generation program to filter out all classes that are not specified in the filter file. This switch takes a single parameter.

Element Information

Number of Occurrences	1.

Parent Elements	`ProductConfiguration`.
Contents	None.

Remarks

Required. Defines the type of analysis and maximum session time.

Example

The following example shows a construction using RuntimeAnalysis following a ProductConfiguration tag. In this example, the Type attribute specifies a performance analysis with a maximum duration of 1000 seconds and suppression of UI error messages.

```
<?xml version="1.0" ?>
<ProductConfiguration xmlns="http://www.microfocus.com/products">
<RuntimeAnalysis Type="Performance" MaximumSessionDuration="1000"
NoUIMsg="true"/>
```

Service Element

Syntax

```
<Service
      CollectData = "true or false"
      Start = "true or false"
      RestartIfRunning = "true or false"
      RestartAtEndOfRun = "true or false"
      RepeatInjection = "true or false"
      >
      ...
</Service>
```

Attributes

CollectData	Optional. Specify `true` or `false`. Defaults to `true` if omitted. Specifies whether profiling will be enabled for the target service.
Start	Optional. Specify `true` or `false`. Defaults to `true` if omitted. Specifies if `DPAnalysis.exe` will start the specified target. If set to `false`, profiling will be enabled for this target but it will not be started; profiling will begin the next time the service is started (by whatever means).
RestartIfRunning	Optional. Specify `true` or `false`. Defaults to `false` if omitted. When you set RestartIfRunning to `true`, `DPAnalysis.exe` will attempt to restart the specified service if it is running on the host computer.
RestartAtEndOfRun	Optional. Specify `true` or `false`. Defaults to `false` if omitted. When you specify `true`, `DPAnalysis.exe` will attempt to restart the service (generating a session file) at the end of the run. If the service is the only process to be profiled, set this attribute to `false`.
RepeatInjection	Optional. Specify `true` or `false`. Defaults to `false` if omitted. Causes `DPAnalysis.exe` to profile the target in every time it runs until you explicitly specify `false`.

Element Information

Number of Occurrences	The configuration file must contain at least `1` `Process` or `1` `Service` element.

Parent Elements	`Targets`.
Contents	AnalysisOptions, Path, Arguments, Working Directory, ExcludeImages, Name, Host.

Remarks

Specifies a target service.

Example

The following example shows a construction using Service within a Targets element.

```
<Targets RunInParallel="true">
  <Service CollectData="true" Start="true" RestartIfRunning="true"
RestartAtEndOfRun="true">
  <Name>ServiceApp</Name>
  </Service>
</Targets>
```

Targets Element

Syntax

```
<Targets RunInParallel="true or false">   ...
</Targets>
```

Attributes

RunInParallel	Optional. Specify `true` or `false`. Defaults to `true` if omitted. If you specify more than one target, defines how the targets are run. When `RunInParallel` is `true`, DPAnalysis starts the target processes and services one right after the other; multiple targets will run at the same time (parallel). Otherwise, DPAnalysis starts target `N + 1` only after process `N` has launched and exited; targets run one at a time (serial).

Element Information

Number of Occurrences	1.
Parent Elements	`RuntimeAnalysis`.
Contents	Process, Service.

Remarks

Required. Begins a block of one or more <`Process`> or <`Service`> entries. Target processes and services are started in the order they are listed in the configuration file.

Example

The following example shows a construction using Targets to specify analysis of one <Service> and two <Process> elements. Note that RunInParallel is **true** so that, for this example, the targets would run in parallel.

```
<Targets RunInParallel="true">
  <Service CollectData="true" Start="true">
    <AnalysisOptions NM_METHOD_GRANULARITY="0" EXCLUDE_SYSTEM_DLLS="1" />
                <Name>ServiceApp</Name>
                <Host>remotemachine</Host>
  </Service>
  <Process CollectData="true" Spawn="true" NoWaitForCompletion="true">
```

```
    <AnalysisOptions NO_MACH5="1" NM_METHOD_GRANULARITY="1" SESSION_DIR="c:
\MyDir" />
    <Path>ClientApp.exe</Path>
    <WorkingDirectory>c:\temp</WorkingDirectory>
  </Process>
  <Process CollectData="true" Spawn="true" NoWaitForCompletion="true">
    <AnalysisOptions NO_MACH5="1" NM_METHOD_GRANULARITY="1" SESSION_DIR="c:
\MyDir" />
    <Path>TestApp02.exe</Path>
    <WorkingDirectory>c:\temp</WorkingDirectory>
  </Process>
</Targets>
```

WorkingDirectory Element

Syntax

```
<WorkingDirectory> c:\MyWorkingDir </WorkingDirectory>
```

Attributes

None.

Element Information

Number of Occurrences	1 per `Process` or `Service`.
Parent Elements	`Process`, `Service`.
Contents	Path to the target directory.

Remarks

Optional. No default if omitted. Set the working directory of the target process or service.

Example

The following example shows a construction using WorkingDirectory nested within a parent Process element.

```
<Process CollectData="true">
<WorkingDirectory>c:\temp</WorkingDirectory>
</Process>
```

Enabling Code Analysis for Execution Plans

To enable code analysis for an execution plan:

1. In the menu, click **Execution Planning** > **Details View**.
2. In the **Execution** tree, select an execution plan.
3. Click the **Deployment** tab.
4. In the **Code Analysis Settings** section of the **Deployment** page, click **Edit**. The **Edit Code Analysis Settings** dialog box displays.
5. Check the **Enable code analysis** check box.
6. Select a profile from the **Code Analysis Profile** list box.
7. In the **Hostnames** text box, enter a comma-separated list of host names, with port, if default port 19129 is not used, from which code analysis information is to be gathered.

   For example `labmachine1, labmachine2:8000, 198.68.0.1`. For each execution plan, you need to define the host names of the machine resources where the AUT is running. For example, with a

client/server system, you must not only gather code coverage information on the client, which probably runs directly on an execution server, but also on the server, which likely runs on a different machine. This applies to all multi-tiered applications.

Note: For JUnit code analysis runs, you do not need to specify a hostname.

8. Click **OK** to save your settings.

Note: Once code analysis has been defined for an execution plan, each future run of that execution plan will gather code coverage information from the defined host names. While monitoring an execution on the **Activities** page, you will see that after gathering the sources for tests, Test Manager gathers full code coverage information before beginning test runs. The Code Coverage Controller, which is integrated into each Test Manager execution server, controls all defined hosts during execution runs. For each test of an execution plan, the controller starts and stops all associated instances, collects XML-based code coverage files for the test, and merges the results into a single file. The test then saves the merged code coverage file to its execution results.

Code Analysis Details Page

Reports > **Code Analysis** > **Details**

The **Details** page displays code-coverage information for selected products, versions, and builds at the product, package, and class levels.

Product level view displays a list of covered and not-covered packages for specific products and product builds. By clicking a package name in **Product** view you can drill down to view code-coverage information for the classes that are included in that package.

Product view displays the following attributes for a selected product:

Attribute	Description
Name	Product name
Statements	Total amount of statements
Packages/Namespaces	Histogram bar view that includes the following figures: • Total percentage of packages/namespaces that are covered • Number of covered packages/namespaces, in green • Number of uncovered packages/namespaces, in red
Classes	Histogram bar view that includes the following figures: • Total percentage of classes that are covered • Number of covered classes, in green • Number of uncovered classes, in red
Methods	Histogram bar view that includes the following figures: • Total percentage of methods that are covered • Number of covered methods, in green • Number of uncovered methods, in red

Package level view displays a list of covered and not-covered classes for specific products and product builds. By clicking a class name in **Package** view you can drill down to view code-coverage information for the methods that are included in that class.

Package view displays the following attributes for each package in the selected product, across multiple rows:

Attribute	Description
Package	Package name
Statements	Total statements
Classes	Histogram bar view that includes the following figures: • Total percentage of classes that are covered • Number of covered classes, in green • Number of uncovered classes, in red
Methods	Histogram bar view that includes the following figures: • Total percentage of methods that are covered • Number of covered methods, in green • Number of uncovered methods, in red

Class level view displays a list of covered and not-covered methods for specific products and product builds.

Class view displays the following attributes for each method, across multiple rows:

Attribute	Description
Name	Method name
Signature	Method signature
Statements	Total statements
Covered	Covered status of method: • `True` indicates that the method is covered • `False` indicates that the method is not covered

Note: When the page includes more elements than can be displayed at once without impacting response time, elements are displayed in increments. Page number links at the bottom of the page allow you to browse through the elements included on the page one page at a time. To display all elements as a single list, click the **[All]** link.

Latest Builds and Build Versions

When you select a product in the navigation tree, the list of packages and classes with coverage information for the latest covered version, which implies the latest covered build for the version, is displayed automatically. When you select a product version in the navigation tree, coverage information for the latest covered build of the version is displayed automatically.

Note: If you are running multiple applications against the same build, the coverage from all of the applications will be merged together.

Note: Code analysis across a range of builds is not supported by Test Manager currently.

Results Compilation

Once an execution plan's test executions are complete, you can view its results. You will notice that there is a new result file for the execution plan called `FullCoverageInfo.xml` and an additional `CodeCoverageInfo.xml` file for each test result. Test Manager uses these result files to aggregate and calculate all code analysis data.

Note: Aggregated data is not immediately available and calculations may take time to compile.

Code Analysis and the Manual Testing Client

The Manual Testing Client supports running test executions with code analysis information. If a test execution has already been enabled to gather code analysis information within Test Manager, the settings are automatically available in the Manual Testing Client. If you want to enable code analysis for an execution plan from within the Manual Testing Client though, see *Enabling Code Analysis with the Manual Testing Client*.

Enabling Code Analysis Within the Manual Testing Client

To enable code analysis for an execution plan from within the Manual Testing Client:

1. In the Manual Testing Client, navigate to **Edit** > **Edit Code Analysis Settings** . The **Edit Code Analysis Settings** dialog box opens.
2. Check the **Enable code analysis for package <Package Name>** check box.

   Note: After code analysis is enabled, you can execute your tests in the Manual Testing Client. However, you need to click **Code Analysis: Start** on the **Details** page of the **Execute Test** dialog box before you actually start testing. This way Test Manager will collect code analysis information while you execute the manual test. When you are done testing, click **Stop** to halt the collection of code analysis information.
3. Select a profile from the **Code Analysis Profile** list box.
4. In the **Hostnames** text box, enter a comma-separated list of hostnames, with port, if default port 19129 is not used, from which code analysis information is to be gathered.

   For example `labmachine1, labmachine2:8000, 198.68.0.1`. For each execution plan, you need to define the host names of the machine resources where the AUT is running. For example, with a client/server system, you must not only gather code coverage information on the client, which probably runs directly on an execution server, but also on the server, which likely runs on a different machine. This applies to all multi-tiered applications.

   Note: For JUnit code analysis runs, you do not need to specify a hostname.
5. Click **OK** to save your settings.

Generating Code-Change Impact Reports

To generate a code-change impact report:

1. In the menu, click **Projects** > **Project List**.
2. Select the project for which you want to analyze code-coverage data.
3. In the menu, click **Reports** > **Code Analysis**.
4. Click **Create Code Change Impact Report** on the toolbar. The **Select Classes for Report** dialog box displays.
5. Select a **Product** and **Version**, if you want to change the pre-selected values.

   Click **Add** to add the class file(s) to the **Selected classes** pick list.

   You can remove classes in the **Selected classes** pick list by selecting entries and clicking **Remove**. Click **Remove All** to remove all selected classes from the **Selected classes** pick list.
6. In the **Filter** text box, type criteria to filter the packages/namespaces. For example, entering the string “published” will only list packages/namespaces that contain the string “published” in their names.
7. Select a package from the **Packages/Namespaces** list box.

   Use **CTRL + CLICK** to select multiple packages/Namespaces.

The classes that are available in the selected package/namespace are displayed in the **Classes** list box.

8. In the **Classes** list box, select a class that you want to have included as a source in your report.

   Use **CTRL + CLICK** to select multiple classes.
9. Click **Add**. The selected classes are added to the **Selected classes** list box.
10. Repeat the preceding steps to add the class files to the **Selected classes** list box until you have added all required classes to the **Selected classes** list.
11. Select a report from the **Select report** list box.
12. Click **OK** to generate the report.

Select Classes for Report Dialog Box

Reports > **Code Analysis** > **Create Code Change Impact Report**

The **Select Classes for Report** dialog box enables you to select class files to be included as sources in a *Code-Change Impact Report*. The dialog box includes the following elements:

Element	Description
Product	Name of the product for which code analysis information is required.
Version	Version of the product for which code analysis information is required.
Filter	Enter criteria for filtering the packages. For example, if you type the string `published`, the *Code-Change Impact Report* displays only packages that contain the string “published” in their names.
Packages/Namespaces	Product packages/namespaces that are specified for coverage in the selected version.
Classes	All classes included in the selected package.
Add	Click to add the selected classes for code-coverage analysis.
Selected Classes	Displays the classes that are already selected for code-coverage analysis.
Remove	Click to remove the selected class from the **Selected Classes** list box.
Remove All	Click to remove all classes from the **Selected Classes** list box.
Select Report	Type of the *Code-Change Impact Report*.

Viewing Code-Coverage Information for Packages/Namespaces

To view code-coverage information for a package/namespace:

1. In the menu, click **Projects** > **Project List**.
2. Select the project for which you want to view code-coverage information.
3. In the menu, click **Reports** > **Code Analysis**.
4. In the **Code Analysis** tree, expand the project node in the navigation tree to display the products that are available for the selected project.
5. Expand a product node to display the versions that are available for that product.
6. Expand a version node to display the builds that are available for that version.

7. Select a specific build. Code coverage information for the selected build displays on the **Details** page.

Note: To view code-analysis information for all products click **Show all products** on the toolbar. Products of other projects are then listed under the **Other Projects** node.

Filtering

Filters

Filters provide an efficient means of finding exactly the information you need, while excluding extraneous detail. Filters highlight only those elements that are relevant to your needs, and enable you to quickly sort through requirements, test elements, and execution plans. By defining global filters, you can create complex filter criteria that are available throughout Test Manager without defining filter criteria each time you need to filter a list.

Based on your needs, you can create new filters, edit existing filters, select filters, delete filters, or turn filtering off at the project level. Projects do not contain default filters. You can access and edit filters from the toolbars in the Test Manager units and from the **Project Settings** unit.

Note: Filters are not applied to reports. The **Recent Changes** filter enables you to view project-wide changes and additions that other users have made to tests since your last change acknowledgement. The **Show Changes/Show All** toggle button and the **Acknowledge** button in the **Tests** area help you to find out what changes other users have made. Your system administrator can configure email notifications that alert you to changes that are made to test settings. Email alerts include links that take you directly to a view of recent changes.

Global Filter Details

Global filters allow you to quickly sort through elements in a Test Manager area, highlighting only those that are relevant to your needs.

To view the details of a global filter, click **Project:<Project Name>** > **Project Settings** > **Filters** in the menu, and click on the name of the filter.

The specific Test Manager area in which a new filter is available is defined by selecting the general filter category. The following list shows the general filter categories:

General Filter Category	Description
Requirement	The filter will be available in the **Requirements** area.
Test	The filter will be available in the **Tests** area
Execution	The filter will be available in the **Execution** area.

Depending on the general filter category, the following attribute categories are available:

Filter Category	Attribute Category
Requirement	• Test Attribute • Test Parameter • Test Property • SilkTest Classic Test Property • Requirements Property • Requirements Coverage • Nested Test Filter • Nested Requirement Filter

Filter Category	Attribute Category
Test	• Test Attribute • Test Parameter • Test Property • SilkTest Classic Test Property • Third Party Test Property • Test Child Nodes • Requirements Property • Requirements Coverage • Assigned Issues • Nested Test Filter • Nested Requirement Filter
Execution	• Execution Property • Execution Schedule • Execution Plan Run • Execution Test Container • Execution Plan Deployment • Nested Execution

Each global filter has to have a property, an operator, and a value from the respective list boxes:

Property Available properties depend on the filter category. It defines the property for which you are defining a filter setting. If you selected an attribute category, the property list includes custom attributes to query against.

Operator Specifies the filter operator. The operator depends on the property type. For example, if you select a property that is based on a string field type, the available operators are = (equals defined value), `not` (differs from the defined value), `contains` (contains the defined value somewhere in the string), and `does not contain` (does not contain the defined value in the string).

Value The value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.

Recent Changes

The *Recent Changes* filter enables you to efficiently view and acknowledge changes and additions that other users have made to requirements, tests, or execution plans project-wide since your last change acknowledgement.

The two buttons at the far-right of the toolbar, the **Show Changes/Show All** toggle button and the **Acknowledge** button, help you to find out what changes other users have made.

Note: Your system administrator can configure email notifications that alert you to changes that are made to test settings. Email alerts include links that take you directly to a view of recent changes.

Creating Filters

To create a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.

2. Click the **Filters** tab to view the list of available filters.
3. Click **New Filter**. The **New Filter** dialog box displays.
4. Type a **Name** for the new filter.

   This name will be displayed in list boxes when the filter becomes available.
5. Select a **Category** for the new filter from the list box to make the filter available in a specific Test Manager area.

Category	Area
Requirement	Requirements
Test	Tests
Execution	Executions

6. Type a **Description** for the new filter.
7. *Optional:* Check the **Visible to other users** check box to allow other users to see the filter.
8. *Optional:* Check the **Editable by other users** check box to allow other users to edit the filter.

   Note: If **Visible to other users** and **Editable by other users** are checked, the filter is public. To delete private filters, you have to be the owner of the filter or you need the **Delete private filters of other users** permission.
9. Select a category of filter criteria from the **Selection criteria** list box. The available categories depend on the general filter category you have selected.

   Note: You can combine filters by selecting **Nested Test Filter** or **Nested Requirements Filter**. Selecting one of these categories allows you to include an existing filter in your new filter.
10. Select a **Property**, **Operator**, and **Value** for the filter from the respective list boxes.

    Property Available properties depend on the filter category that you have selected in the previous step. Defines the property for which you are defining a filter setting. If you have selected an attribute category, the property list includes custom attributes to query against.

    Operator Specifies the filter operator. The operator depends on the property type you have selected. For example, if you have selected a property that is based on a string field type, the following operators are available:

Operator	Description
=	The string equals the defined value.
not	The string is different than the defined value.
contains	The string contains the defined value.
not contains	The string does not contain the defined value.

    Value Enter the value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.
11. *Optional:* Click **More** if you want to add more than one filter category to the new filter. Repeat this procedure to define new categories.

    Note: If you define more than one filter category, you must define whether the categories need to be fulfilled in addition to the existing categories (AND relationship), or if the filter returns true when one of the filter categories is fulfilled (OR relationship). Select either **AND** or **OR** to define the filter category relationship. You cannot define nested AND, OR relationships.
12. *Optional:* To remove filter categories, click **Fewer**. This removes the last filter category.
13. Click **OK** to save the new filter, or click **Cancel** to abort the operation.

Creating Advanced Filters

Advanced custom filters enable you to combine simple filters to create complex filters that apply multiple filter criteria simultaneously.

To create an advanced custom filter:

1. In the menu, select the appropriate area, **Execution Planning**, **Requirements**, or **Tests**.
2. Create a new custom filter.

   For additional information, see *Creating Filters*.
3. Click **More** to display a second set of filter-parameter fields with which you can define a second set of filter parameters.
4. Select a logical operator for the application of the filtering queries. For example, with the operator `and` filtered elements must meet both sets of criteria and with the operator `or` filtered elements must meet one, but not both, of the criteria sets.
5. To delete a filter-parameter string, click .
6. To display additional filter-parameter fields and create additional filter queries, click **More**. To remove excess filter-parameter sets, click **Fewer**.

Applying Filters

After you have created and stored a custom filter, you can apply that filter to the selected tree. Custom filters can be applied for requirements, tests and execution plans. Only elements that meet applied filter criteria are displayed in the tree.

Note: Filtered requirements are returned in read-only form and can not be edited. The **Edit Properties** button is disabled for filtered requirements.

To apply a stored filter:

1. In the menu, select the appropriate area, **Execution Planning**, **Requirements**, or **Tests**.
2. Select the desired filter from the **Filter** list box on the toolbar.
3. All elements that meet the filter's criteria are displayed.

   Note: To remove filtering and display all elements, select **<No Filter>** from the **Filter** list box on the toolbar.

Editing Filters

To edit a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Filters** tab to view the list of available filters.
3. Click the name of the filter you want to edit. The **Edit Filter** dialog box displays.
4. Edit the **Name** and **Description** of the filter.
5. Select a category of filter criteria from the **Selection criteria** list box. The available categories depend on the general filter category you have selected.

   Note: You can combine filters by selecting **Nested Test Filter** or **Nested Requirements Filter**. Selecting one of these categories allows you to include an existing filter in your new filter.
6. Select a **Property**, **Operator**, and **Value** for the filter from the respective list boxes.

Property Available properties depend on the filter category that you have selected in the previous step. Defines the property for which you are defining a filter setting. If you have selected an attribute category, the property list includes custom attributes to query against.

Operator Specifies the filter operator. The operator depends on the property type you have selected. For example, if you have selected a property that is based on a string field type, the following operators are available:

Operator	Description
=	The string equals the defined value.
not	The string is different than the defined value.
contains	The string contains the defined value.
not contains	The string does not contain the defined value.

Value Enter the value that you want to filter out. Depending on the property type that you have selected, values will either be strings that you can enter into the text box, or a selection of predefined values that you can select from the list box.

7. Click **OK** to save the edited filter definition.

Copying Filters

Create a new filter by copying an existing one and changing the settings.

1. In the menu, select the appropriate area, **Execution Planning**, **Requirements**, or **Tests**.
2. In the toolbar of the selected area, select a filter from the list box.
3. Click . The **Copy Filter** dialog box opens with the settings of the original filter.
4. Edit the properties of the filter and click **OK**.

Deleting Filters

To delete a filter:

1. In the menu, click **Project:<Project Name>** > **Project Settings** .

   Note: If you have not selected a project, a warning message opens, asking you to select a project. Select the project for which you want to define the setting.
2. Click the **Filters** tab to view the list of available filters.
3. Click **Delete Filter** for the filter that you want to delete. A confirmation dialog box displays, asking you you to confirm the deletion.
4. Click **Yes** to delete the selected filter or **No** to abort the operation. If you click **Yes**, you will be returned to the filters list and the deleted filter is no longer displayed.

System Diagnostics

Users can use the system diagnostics page to provide all relevant support information with one click.

The page contains the following information:

- Product version.
- Version and type of database.
- Used integrations.
- System environment information and system properties.
- JDBC information.

- Database statistics: number of projects, test types, indices, triggers, constraints.
- Application server and front-end server logs.

The page can be zipped and downloaded to the local file-system by clicking the **Download** button. You can select the server logs that should be downloaded by selecting them via the check boxes.

Note: In order to view the page, a user must have **View system settings** rights.

Viewing System Diagnostics

In order to view the page, a user must have **View system settings** rights.

1. Click **Help** > **About**.
2. Click **System Diagnostics**. A login dialog displays.
3. Enter the login of an existing SilkCentral Test Manager user, who at least has the privilege **View system settings**.
4. Select the SilkCentral Test Manager server logs that should be downloaded by selecting them via the check-boxes.
5. Click **Download** to zip and download the data to the local file-system.

Index

.NET Explorer
    configuring 159

A

aborting
    manual test executions 253
accessing
    context-sensitive execution reports 364
    context-sensitive execution-plan-run reports 364
    context-sensitive requirements reports 365
    context-sensitive test reports 365
    most recently used reports 392
    tests from grid view 165
activities
    across projects 292
    managing 284
    page 288
Activities page
    columns, displaying and hiding 284
    columns, reordering 287
    columns, resizing 287
    default settings 287
    deleting last execution runs 284
    entering issues 285
    filtering boolean-based values 286
    filtering date-based values 285
    filtering number-based values 286
    filtering predefined list 286
    filtering test runs 285
    filtering text-based values 285
    filters, removing 287
    grouping 286
    grouping test runs, overview 286
    remove grouping 286
    sorting test runs 287
adding
    call to shared steps 18, 188
    data source values to manual test steps 184
    definite runs 248
    dependent execution plans 35, 239
    exclusions 247
    manual testers 233
    manual testers to configurations 222
    multiple TestPartner tests 158, 355
    parameters to configurations 220
    parameters to tests 39, 175
    test containers 145
    test folders 147
adding AUT host
    SilkTest Classic 233, 353
adding internal issues
    Manual Testing Client 265
adding issue tracking profiles
    Bugzilla 71, 302
    Changepoint 77, 303
    IBM Rational ClearQuest 72, 305
    JIRA 69, 300
    Rational ClearQuest 72, 305
    SilkCentral Issue Manager 65, 306
    StarTeam 67, 308
    Team Foundation Server 74, 311
    TFS 74, 311
adding links
    test containers 146
adding quality goals 294
adding source control profiles
    CVS 83, 324
    MSVSS 85, 328
    PVCS 80, 330
    SilkTest Workbench 92, 331
    StarTeam 82, 333
    subversion 87, 334
    SVN 87, 334
    TestPartner 79, 336
    TFS 94, 326
    UNC 89, 338
    Version Manager 80, 330
    VFS 91, 323
    Visual SourceSafe 85, 328
adding TFS issue tracking profiles
    installing proxy service 74, 310
advanced context-sensitive execution reports
    enabling 366
advanced context-sensitive requirement reports
    enabling 366
advanced context-sensitive test reports
    enabling 367
agent computers
    overview 14
agile planning tools
    integration default nodes 136
analyzing
    SilkPerformer test results 281, 346
    test runs 269
analyzing test results
    quick start task 22
application server
    overview 14
applying
    filters 422
architecture
    overview 14
assigned execution plans
    overview 181
assigned executions
    viewing 181
assigned manual tests
    print 253
assigned requirements
    locating 178
    page 179
    removing 179
assigned requirements page
    tests 179

assigned tests
print manual tests 253
removing from requirements 110
assigned tests page
executions 243
requirements 111
assigning
external issues 298
keywords to configurations 221
keywords to execution plans 36, 236
requirements to tests 40, 178
tests from grid view to execution plans 33, 242
tests to attributes 38, 173
tests to execution plans 242
tests to execution plans manually 33, 242
tests to execution plans through filter 34, 242
assigning issues to manual test steps 171
assigning manual tests
to manual testers 210
assigning requirements to tests
overview 178
assigning tests
execution plans 242
to testing cycles 210
Atlassian JIRA
issue tracking profiles 69, 300
attached images
viewing in Manual Testing Client 263
attaching
files to requirements 29, 107
files to test elements 40, 179
links to requirements 29, 108
links to test elements 41, 180
attaching files
requirements 29, 107
attaching links
requirements 29, 108
attachments
requirements 107
tests 179
viewing in Manual Testing Client 263
attachments page
requirements 109
attribute types 51, 56
attributes
assigning to tests 38, 173
creating 20, 50
deleting 51
deleting from tests 174
editing 51, 174
overview 50
page 52
tests 173
attributes page
tests 174
automated tests
manual execution 251
average page-time trend
reports 347, 372
average transaction busy-time trend
reports 348, 373

B

baselines
CaliberRM 126, 318
comparing with projects 45
BIRT
customizing report templates 26, 390
customizing reports 390
browser
settings 12
Bugzilla
adding issue tracking profiles 71, 302
editing issue tracking profiles 72, 302
issue tracking profiles 71, 301
build information
projects 44
updates 44
build information files
creating 44
updates 44

C

calculated properties
about 53
add-edit page 54
creating 53
calendar
specifying range 299
calendar tool 299
CaliberRDM
integrating 129, 316
integration 61
integrations 316
CaliberRM
baselines 126, 318
copying integrated projects 126, 319
creating baseline 127, 319
handling test assignments 126, 319
integrating 125, 317
integration 61
call to shared steps
adding 18, 188
dialog box description 189
calling
test steps 188
capturing
tests 295
change notification
enabling 22, 58
list of changes 59
overview 58
change notifications
disabling 58
page 59
requirements 115
Changepoint
adding issue tracking profiles 77, 303
issue tracking profiles 76, 303
changes
requirements 114
tests 151

changing
test status 269
chart server
overview 14
charts
displaying 27, 389
overview 389
printing 389
removing 389
child requirements
creating 100
cleaning up
test packages 150
clearing
parameter assignments 176
clearing flags
requirements 118
code analysis
build versions 416
configuration 400
configure DPAnalysis to launch your AUT 399
details page 415
enabling 397
generating code-change impact reports 27, 417
Java examples 397
java known limitations 397
Java options 397
java prerequisites 397
latest builds 416
Manual Testing Client 417
overview 396
results compilation 416
running manual tests 400
select classes for report dialog 418
viewing coverage information for packages/ namespaces 418
windows code analysis framework and devpartner analytics 399
code analysis and automated testing
multiple AUTs 401
one AUT 401
overview 400
code analysis examples
Java 397
code coverage
overview 396
reports 362
trend report 362
code coverage reports
method coverage comparison 362
trend 362
code-change impact
reports 361
code-coverage information
viewing for packages/namespaces 418
collapsing
execution plans tree 225
tests tree 135
columns
resizing in grid view 167
comparing
projects and baselines 45
versions 19, 193
configuration suite
configuration page 223
configuration suites
copying 225
deleting 225
document view 224
editing 225
managing 228
configuration testing
adding manual testers 222
adding parameters to configurations 220
assigning keywords to configurations 221
creating configuration suites from execution plans 220
creating suite 220
overview 219
removing keywords 222
removing parameters from configurations 221
removing testers 222
configurations page 223
configuring
.NET Explorer 159
deployment environments 233
execution dependencies 239
FitNesse 164
JUnit 159, 339
Manual Testing Client 261
Manual Testing Client connection parameters 261
Manual Testing Client package upload preferences 262
Manual Testing Client settings 262
manual tests 160
MSTest 164, 341
NUnit 161, 341
process executor 163, 342
requirement types 28, 101
requirements management 60
SAP Solution Manager 314
SilkPerformer 161, 343
SilkTest Classic 162
SilkTest Classic plan 158
SilkTest Workbench 162, 352
test properties 156
TestPartner 158, 354
Windows scripting 163, 355
configuring client locations
MSVSS 87, 329
configuring projects
quick start task 20
settings 20, 47
Telelogic DOORS 128, 320
connection parameters
configuring 261
contents
tests 138
context-sensitive execution reports
accessing 364
enabling 365
context-sensitive execution-plan-run reports
accessing 364
context-sensitive reports
accessing execution 364

accessing execution-plan run 364
accessing requirements 365
accessing test 365
enabling advanced execution 366
enabling advanced requirement 366
enabling advanced test 367
enabling execution 365
enabling requirement 366
enabling test 366
context-sensitive requirement reports
enabling 366
context-sensitive requirements reports
accessing 365
context-sensitive test reports
accessing 365
enabling 366
copy to other projects
test elements 135
copying
CaliberRM integrated projects 126, 319
configuration suites 225
execution plan folders 225
execution plans 225
filters 423
test elements 135
test packages 150
coverage
direct mode 113
full mode 113
modes 113
requirements 112
coverage page
requirements 113
creating
advanced filters 31, 422
build information files 44
child requirements 100
configuration suite 220
configuration suites from execution plans 220
custom parameters 175
custom requirement properties 55
custom schedules 34, 246
custom step properties 22, 57
data-driven tests 38, 183
execution plan parameters 230
execution plans 32, 228
execution plans in grid view 165, 229
filters 21, 30, 39, 48, 420
issues 296
keywords 235
mashup tab 204, 312
reports 22, 383
requirements 28, 100
shared steps 18, 189
step properties 22, 57
test packages 37, 148
tests 37, 155
creating baseline
CaliberRM 127, 319
creating filters
flagged requirements 118
creating manual tests
outside user interface 171
creating reports
overview 383
creating testing cycles 208
CSV
configuring data sources 97
CSV data
downloading from a data source 184
current run
page 276
custom attributes
creating 20, 50
deleting 51
editing 51
overview 50
custom measure trend
reports 349, 374
custom reports
SQL functions 385
custom requirement properties
creating 55
delete 56
page 52
custom schedules
creating 34, 246
custom step properties
creating 22, 57
deleting 57
editing 57
page 58
customizing report templates
BIRT 26, 390
customizing reports
overview 390
cutting
test elements 135
CVS
adding source control profiles 83, 324
editing source control profiles 84, 325
source control profiles 83, 324

D

dashboard
adding panels 41
overview 41
dashboard panels
introduction 42
issues created per tester 42
manual tests assigned to me 42
requirements coverage status 42
testing activity 42
testing cycle progress 42
data source values
adding to manual test steps 184
data sources
configuration page 99
configuring CSV 97
configuring Excel 97
configuring JDBC 96
data-driven tests 96
deleting 99

donwloading Excel files 98
page 99
synchronizing 98
uploading updated Excel files 98
data-driven
creating tests 38, 183
data set page 185
downloading CSV data 184
editing properties 184
data-driven properties
removing 184
data-driven tests
data import considerations 185
data sources 96
Excel 185
instances 185
overview 183
test types 185
viewing activities 273
worksheet handling 185
database server
overview 14
definite runs
adding 248
deleting 248
editing 248
schedules 248
delete
requirement properties 56
deleting
attributes from tests 174
configuration suites 225
custom step properties 57
data sources 99
definite runs 248
exclusions 247
execution plan dependencies 240
execution plan folders 225
execution plan results 270
execution plans 225
filters 50, 423
issue tracking profiles 78
issues 298
obsolete requirements 102
requirement attachments 108
source control profiles 95
step properties 57
sub-reports 387
test attachments 180
test elements 135
test run results 270
deleting property mapping
requirements integration 133
deleting quality goals 294
deleting testing cycles 211
dependencies
deleting 240
editing 240
page 241
deployment
page 237
deployment environments
configuring 233
details view
viewing issue statistics 297
devpartner
dpanalysis.exe 402
DevPartner Code Coverage Integration 300
disabling
change notifications 58
requirements integration 133
displaying
charts 27, 389
displaying columns
grid view 166
document view
requirements 104
tests 134
viewing issue statistics 297
donwloading Excel files
data sources 98
downloading
CSV data from a data source 184
report templates 388
SilkPerformer projects 279, 344
SilkPerformer test result packages 282, 346
Upload Manager 152
dpanalysis.exe configuration file
analysisoptions element 405
arguments element 407
excludeimages element 408
host element 408
name element 409
path element 409
process element 410
runtimeanalysis element 411
service element 412
targets element 413
workingdirectory element 414
dynamic hardware provisioning
keywords 234

E

edit
library visibility 188
editing
attributes 174
configuration suites 225
custom requirement properties 56
custom step properties 57
data-driven properties 184
definite runs 248
dependencies 240
execution plan folders 225
execution plans 225
external properties 131
filters 49, 422
manual test steps 18, 171, 189
parameters 176
report parameters 25, 394
report properties 24, 393
requirements 101
SilkPerformer test properties 281, 345

step properties 57
success conditions 145
test containers 147
test folders 147
test mapping file 203
tests 37, 155
editing issue tracking profiles
Bugzilla 72, 302
IBM Rational ClearQuest 73, 305
JIRA 70, 301
Rational ClearQuest 73, 305
SilkCentral Issue Manager 66, 307
StarTeam 68, 309
Team Foundation Server 75, 311
TFS 75, 311
editing manual test steps
Test Manager 18, 171, 189
editing mapping file
importing requirements 122
editing property mapping
requirements integration 132
editing source control profiles
CVS 84, 325
MSVSS 86, 329
PVCS 81, 330
SilkTest Workbench 93, 332
StarTeam 83, 333
Subversion 88, 335
SVN 88, 335
TestPartner 80, 337
TFS 94, 327
UNC 90, 338
Version Manager 81, 330
VFS 92, 324
email notifications
automatic requirement synchronization 124
enabling
advanced context-sensitive execution reports 366
advanced context-sensitive requirement reports 366
advanced context-sensitive test reports 367
change notification 22, 58
code analysis 397
code analysis within Manual Testing Client 265, 417
context-sensitive execution reports 365
context-sensitive requirement reports 366
context-sensitive test reports 366
enabling code analysis
execution plans 414
Excel
configuring data sources 97
downloading from data source 98
uploading updated files 98
Excel import
versioning tests 203
Excel update
versioning tests 203
exclusions
adding 247
deleting 247
editing 247
schedules 247
execute test
dialog box 266
executing
attended SilkPerformer tests 280, 345
automated tests manually 251
individual execution plans 36, 250
manual tests 252
tests 250
executing manual tests
current run page 252
user interface 252
executing SilkTest Classic tests
overview 283, 353
execution
manual execution planning 206
schedules page 248
setup/cleanup page 232
execution dependencies
configuring 239
execution packages
downloading to Manual Testing Client 263
editing build numbers 264
exporting from Manual Testing Client 263
importing into Manual Testing Client 264
overview 263
execution parameters
page 231
execution plan folders
copying 225
deleting 225
editing 225
execution plan properties
document view 224
execution plan runs
details 270
execution planning
adding manual testers to a testing cycle 209
dependencies page 241
edit assigned testers 209
overview 205
reports 367
toolbar functions 205
execution plans
adding dependent 35, 239
adding manual testers 233
adding parameters 230
assigning keywords 36, 236
assigning tests 242
assigning tests from grid view 33, 242
assigning tests through filter 34, 242
combinations of automated and manual tests 231
configuring cleanup executions 34, 232
configuring setup executions 34, 232
copying 225
creating 32, 228
creating configuration suites 220
creating in grid view 165, 229
deleting 225
deleting dependencies 240
deleting results 270
dynamic hardware provisioning 234
editing 225
editing dependencies 240

enabling code analysis 414
executing 36, 250
locating assigned tests 243
managing 228
manually assigning tests 33, 242
not passed tests 370
overriding test parameters 230
parameters 230
print assigned manual tests 253
removing assigned tests 243
removing keywords 236
removing parameters 230
removing testers 234
run comparison reports 367
run details 270
run errors report 368
schedules 245
status calculation 229
status overview 371
tree 225
execution plans tree
collapsing 225
expanding 225
filtering 226
execution reports
not passed tests 370
run comparison 367
run errors 368
status overview 371
execution server
overview 14
execution status
overview report 371
executions
assigned tests page 243
current run page 276
deployment page 237
notifications page 241
parameters page 231
properties page 226
run dialog box 250
runs page 273
expanding
execution plans tree 225
tests tree 135
exporting
tests to Excel 200
External IDs
test packages 148
external properties
editing 131
overview 131
viewing 132
external requirements
generating tests 131, 317
external requirements management tools
overview 60, 122
external systems
communicating over SSL 360

F

files
attaching to test elements 40, 179
filtering
boolean values in grid view 168
date-based values in grid view 167
execution plans tree 226
number-based values in grid view 168
overview 419
requirements tree 104
test containers 136
test folders 136
tests in grid view 167
text-based values in grid view 167
values using predefined list in grid view 168
filtering test runs
Activities page 285
filters
applying 422
assigning tests to execution plans 34, 242
copying 423
creating 21, 30, 39, 48, 420
creating advanced 31, 422
deleting 50, 423
details 419
editing 49, 422
global 47, 419
overview 47, 419
page 50
recent changes 420
removing from grid view 168
removing specific from grid view 168
finding
requirement properties 106
test properties 156
finding assigned tests
requirements 110
FitNesse
configuring 164
flagged requirements
creating filters 118
flags
requirements 118
folders
managing 228
front-end server
overview 14

G

generating reports
code-change impact 27, 417
generating tests
external requirements 131, 317
global filters
overview 47, 419
global schedules
specifying 246
grid view
accessing tests 165
assigning tests to execution plans 33, 242
assigning tests to requirements 109
creating execution plans 165, 229
displaying and hiding columns 166
filtering boolean values 168

filtering date-based values 167
filtering number-based values 168
filtering tests 167
filtering text-based values 167
filtering values using predefined list 168
grouping tests 166
multi-edit 166
overview 165
removing all filters 168
removing specific filters 168
reordering columns 167
resizing columns 167
restoring default settings 169
sorting tests 166
tests 165
grid view filters
removing all 168
removing specific 168
grouping
tests in grid view 166

H

handling test assignments
CaliberRM 126, 319
hiding columns
grid view 166
history
requirements 114
tests 151
history page
requirements 115
tests 151

I

IBM Rational ClearQuest
adding issue tracking profiles 72, 305
editing issue tracking profiles 73, 305
issue tracking profiles 72, 304
IBM Rational Requisite Pro
integrating 127, 315
IBM Rational RequisitePro
integration 61
importing
tests from Excel 200
importing requirements
mapping file 120
Word 119
importing tests from Excel
mapping file 201
test file 201
installing
Manual Testing Client 259
Team Foundation Server Web Service Proxy 74, 310
Telelogic DOORS client on front-end server 128, 320
Test Manager 12
Upload Manager 152
installing proxy service
TFS 74, 310
integrating
CaliberRDM 129, 316
CaliberRM 125, 317
IBM Rational Requisite Pro 127, 315
Rally 130, 203, 312, 322
Telelogic DOORS 128, 320
integrating CaliberRM
overview 125, 317
integrating Rally
overview 130, 321
integration
CaliberRDM 61
CaliberRM 61
IBM Rational RequisitePro 61
SAP Solution Manager 314
Telelogic DOORS 61
integration default nodes
tests 136
integrations
CaliberRDM 316
process executor 342
internal property values 55
issue reports
issues per component 372
issue tracking
page 78
issue tracking profiles
Atlassian JIRA 69, 300
Bugzilla 71, 301
Changepoint 76, 303
deleting 78
IBM Rational ClearQuest 72, 304
JIRA 69, 300
mapping issue states 65
overview 64
page 78
Rational ClearQuest 72, 304
SAP Solution Manager 314
SilkCentral Issue Manager 65
StarTeam 67, 308
Team Foundation Server 74, 310
TFS 74, 310
issues
Activities page 285
assigning external 298
creating 296
deleting 298
document view 297
overview 296
page 298
synchronizing internal and external states 298
viewing statistics in details view 297
viewing statistics in document view 297
issues per component
reports 372

J

Java options
code analysis 397
Java system properties
accessing Test Manager parameters 340
JDBC
configuring data sources 96

JIRA
adding issue tracking profiles 69, 300
editing issue tracking profiles 70, 301
issue tracking profiles 69, 300
JUnit
configuring 159, 339
JUnit tests
available Test Manager parameters 340

K

keywords
assigning to configurations 221
assigning to execution plans 36, 236
creating 235
dynamic hardware provisioning 234
removing from configurations 222
removing from execution plans 236

L

last executions
deleting 284
libraries
change track 194
concept 186
display in projects 187
history 194
introduction 186
overview 186
properties 187
step properties 187
topics 186
visibility 187
library visibility
edit 188
licensing
Test Manager 12
linking
Rally user stories to tests 204, 313
links
attaching to test elements 41, 180
lists
project baselines 46
projects 46
locating
assigned requirements 178
logging in and out 12

M

managing
requirements 100
managing attachments
Manual Testing Client 262
managing requirements
quick start task 28
managing shared step libraries
quick start task 18
managing test executions
quick start task 32
managing tests
quick start task 37
manual execution planning
assigning tests to manual testers 219
matching tests 207
overview 206
specify configurations 215
test assignment 212
test selection 206
manual execution planning - walkthrough
adding manual testers 217
creating testing cycles 216
selecting tests and scheduling 218
user interface 215
walkthrough 215
manual executions
planning 206
manual test executions
calculating status 253
manual test steps
adding data source values 184
assigning issues to 171
custom properties page 58
editing 18, 171, 189
properties page 58
manual testers
adding to configurations 222
assigning manual tests to 210
moving tests to other manual testers 213
manual testing
recording videos in the web ui 254
video capture in the web ui 254
Manual Testing Client
adding internal issues 265
code analysis 417
configuring 261
configuring connection parameters 261
configuring package upload preferences 262
configuring settings 262
downloading execution packages 263
editing package build numbers 264
enabling code analysis 265, 417
exporting execution packages 263
importing execution packages 264
inserting parameters 261
installing 259
managing attachments 262
managing change conflicts 264
overview 14, 254
pasting screen captures 262
starting 259
storing test results 265
tests 260
tour of the UI 254
uninstalling 260
uploading attachments 263
uploading test results 264
video capturing 268, 269
viewing attached images 263
viewing attachments 263
working offline 266
manual tests
aborting 253
adding calls to shared steps 18, 188

assigning to manual testers 210
calls 188
configuring 160
converting to automated 172
converting to automated tests 172
creating outside user interface 171
executing 252
executing in current run page 252
executing through Web interface 252
Manual Testing Client 260
overview 169
printing 172
step properties 187
versions 192
mapping file
editing 122
importing requirements 120
importing tests from Excel 201
mapping issue states
issue tracking profiles 65
mashup tab
creating 204, 312
method coverage comparison
reports 362
moving tests
from one manual tester to another 213
from one testing cycle to another 212
MSTest
configuring 164, 341
MSVSS
adding source control profiles 85, 328
configuring client locations 87, 329
editing source control profiles 86, 329
source control profiles 85, 327
multi-edit
test attributes in grid view 166
test properties in grid view 166
multiple TestPartner tests
adding 158, 355

N

no schedules
specifying 247
notations
parameters 176
notifications
page 59, 241
NUnit
configuring 161, 341

O

obsolete
requirements 102
obsolete requirements
convert to active 102
deleting 102
Office import
editing the test mapping file 203
exporting tests 200
importing tests 200
importing tests from Excel 200
test Excel file 201
test mapping file 201
updating tests 200
open interface
creating manual tests 171
opening
SilkPerformer projects 280, 344
overall page-time trend
reports 350, 375
overall transaction busy-time trend
reports 351, 376
overriding
test parameters 230
overview
product 6
test folders 147

P

package build numbers
editing in Manual Testing Client 264
packages/namespaces
viewing code-coverage information 418
panels
adding to dashboard 41
parameters
adding to configurations 220
adding to execution plans 230
adding to tests 39, 175
calling from JUnit tests 340
clearing 176
creating custom 175
editing 176
inserting 261
notations 176
overriding test parameter 230
page 177
parameters within parameter values 176
removing from configurations 221
removing from execution plans 230
replacing upon execution 177
pasting
screen captures to results 262
test elements 135
performance trend
reports 347, 372
performance trend reports
average page-time 347, 372
average transaction busy-time 348, 373
custom measure 349, 374
overall page-time 350, 375
overall transaction busy-time 351, 376
printing
charts 389
manual tests 172
process executor
configuring 163, 342
integrations 342
product
overview 6
project baselines

lists 46
project settings
page 47
projects
accessing recent 46
adding quality goals 294
build information 44
build information files 44
comparing with baselines 45
configuring settings 20, 47
deleting quality goals 294
lists 46
overview 44
overview report 396
quality goals 293
quality goals page 293
see libraries 187
selecting 45
setting active 45
settings 47
properties
libraries 187
properties page
requirements 106
PVCS
adding source control profiles 80, 330
editing source control profiles 81, 330
source control profiles 80, 330

Q

quality goals
risk-based testing 16
quality goals execution report 380
quality goals planning report 379
quick start tasks
overview 17

R

Rally
creating mashup 204, 312
integrating 130, 322
integrating into 203, 312
linking user stories to tests 204, 313
Rally user stories
linking to tests 204, 313
Rational ClearQuest
adding issue tracking profiles 72, 305
editing issue tracking profiles 73, 305
issue tracking profiles 72, 304
re-run
not executed tests 250
recent changes
filters 420
recent projects
accessing 46
switching 46
recording videos
Manual Testing Client 268
removing
assigned requirements 179
assigned tests from requirements 110
charts 389
data-driven properties 184
execution plan parameters 230
keywords from configurations 222
keywords from execution plans 236
parameters from configurations 221
report templates 388
requirements integration 133
testers from configurations 222
testers from execution plans 234
removing tests
from manual testers 214
from testing cycles 213
reordering
columns in grid view 167
test elements 136
replacing
requirement properties 105
test properties 157
report parameters
editing 25, 394
overview 394
report properties
editing 24, 393
overview 393
report templates
downloading 388
overview 387
removing 388
uploading 387
reports
accessing MRU 392
adding sub-reports 26, 387
all related issues 117, 378
average page-time trend 347, 372
average transaction busy-time trend 348, 373
bookmarking 391
chart page 390
code coverage 362
code coverage trend 362
code-change impact 361
context-sensitive 363
creating 22, 383
creating and customizing 383
custom measure trend 349, 374
data page 395
default reports 361
deleting sub-reports 387
displaying charts 27, 389
downloading templates 388
editing parameters 25, 394
editing properties 24, 393
execution plan run comparison 367
execution plan run errors 368
execution planning 367
execution status overview 371
generating 26, 391
generating code-change impact 27, 417
issues per component 372
manual tests 142, 380
method coverage comparison 362

not passed tests per execution plan 370
overall page-time trend 350, 375
overall transaction busy-time trend 351, 376
overview 361
parameters page 395
performance trend 347, 372
printing charts 389
project overview 293, 376, 396
properties page 393
removing charts 389
removing templates 388
report page 395
reporting area 390
requirement documents 116, 378
requirement progress 116, 377
requirement status 115, 377
requirements 115, 376
sample 383
saving 391
test baseline comparison 142, 381
test progress 141, 380
test run comparison 142, 381
test status 141, 379
tests 141, 379
toolbar functions 392
uploading templates 387
viewing 26, 391
viewing as PDF 391
writing advanced SQL queries 25, 386
requirement attachments
deleting 108
overview 107
viewing 108
requirement properties
custom 55
delete 56
editing 56
finding 106
requirement property types 51, 56
requirement reports
all related issues 117, 378
documents 116, 378
progress 116, 377
status 115, 377
requirement types
configuring 28, 101
requirements
activate obsolete 102
all related issues report 117, 378
assigned tests page 111
assigning tests from grid view 109
attaching files 29, 107
attaching links 29, 108
attachments 107
attachments page 109
calculated properties 53
change notifications 115
changes 114
clearing flags 118
collapsing tree 104
configuring types 28, 101
coverage 112
coverage modes 113
coverage page 113
creating 28, 100
creating a calculated property 53
creating child requirements 100
creating custom properties 55
custom properties 55
custom properties page 52
delete properties 56
deleting attachments 108
deleting obsolete 102
document reports 116, 378
document view 104
editing 101
editing external properties 131
editing mapping file 122
email notifications for automatic synchronization 124
expanding tree 104
finding properties 106
flags 118
generating tests 29, 31, 110
history 114
history page 115
internal property values 55
locating assigned tests 110
managing 100
manually assigning tests 110
marking as obsolete 102
overview 100
progress reports 116, 377
properties 105
properties page 106
removing assigned tests 110
replacing custom properties 105
reports 115, 376
setting flags 118
status reports 115, 377
synchronizing 122
synchronizing across tools 124
synchronizing based on schedules 124
test coverage status 113
toggling coverage modes 113
toolbar functions 102
tracking history 114
tree 104
updating from Word 120
viewing attachments 108
viewing external properties 132
viewing recent changes 114
Word file 120
working with tests 109
requirements coverage
modes 113
requirements import
overview 119
requirements integration
deleting property mapping 133
disabling 133
editing property mapping 132
removing 133
requirements management
page 61

requirements mapping file
editing 122
requirements tree
collapsing 104
expanding 104
filtering 104
requirments management
configuring 60
rerun
not executed tests 250
resetting
grid view 169
resizing
columns in grid view 167
restoring default settings
grid view 169
result files
WSH 195, 357
results
screenshots 295
videos 295
reuse
test steps 18, 189
reusing
test packages 150
reverting
test packages to tests 150
risk based testing
about quality goals 293
adding quality goals 294
deleting quality goals 294
quality goals execution report 380
quality goals planning report 379
quality goals reference page 293
risk-based testing
overview 15
quality goals 16
run
dialog box 250
runs
page 273

S

SAP Solution Manager
configuring 314
issue tracking profiles 314
system requirements 314
SAP Solution Manager integration 314
saving
reports 391
schedules
adding definite runs 248
adding exclusions 247
creating custom 34, 246
definite runs 245, 248
deleting definite runs 248
deleting exclusions 247
editing definite runs 248
exclusions 245, 247
specifying global 246
specifying no schedules 247
scheduling
execution plans 245
screen capturing
enabling for tests 296
setting flags
requirements 118
settings
attribute types 51, 56
attributes 50
attributes page 52
browser 12
change notification 58
creating custom requirement properties 55
filters 47, 419
filters page 50
overview 46
projects 47
requirement property types 51, 56
shared steps
creating 18, 189
detaching 189
shared steps objects
adding calls 18, 188
calls 188
overview 188
steps 169, 190
test steps 169, 190
usages 192
versions 192
visibility 187
Silk4J 339
Silk4NET 341
SilkCentral
access 15
architecture 14
licensing 15
SilkCentral Administration Module
overview 13
SilkCentral file pool
uploading files 152
SilkCentral Issue Manager
adding issue tracking profiles 65, 66, 306, 307
issue tracking profiles 65
overview 12, 306
SilkMeter license server
overview 14
SilkPerformer
analyzing test results 281, 346
configuring 161, 343
downloading test result packages 282, 346
editing test properties 281, 345
executing attended tests 280, 345
Performance Explorer 13, 351
projects 13, 279, 343
uploading test results 282, 347
SilkPerformer projects
downloading 279, 344
opening 280, 344
SilkTest Classic
adding AUT host 233, 353
agent under test 283, 353
automated execution of data-driven testcases 283, 353
configuring 162
executing automated tests 283, 353

logs 283, 354
time-out settings 283, 354
SilkTest Classic plan
configuring 158
SilkTest Classic tests
special characteristics 172
SilkTest Workbench
adding source control profiles 92, 331
configuring 162, 352
editing source control profiles 93, 332
source control profiles 92, 331
test parameters 178, 352
software quality optimization 6
sorting
test folders 148
tests in grid view 166
source control
page 95
source control profiles
CVS 83, 324
deleting 95
MSVSS 85, 327
overview 78
page 95
PVCS 80, 330
SilkTest Workbench 92, 331
StarTeam 82, 333
Subversion 87, 334
SVN 87, 334
Team Foundation Server 94, 326
TestPartner 79, 336
TFS 94, 326
UNC 89, 337
Universal Naming Convention 89, 337
Version Manager 80, 330
VFS 90, 322
Virtual File System 90, 322
Visual SourceSafe 85, 327
specifying
calendar range 299
global schedules 246
no schedules 247
SSL
handling self-signed certificates 360
StarTeam
adding issue tracking profiles 67, 308
adding source control profiles 82, 333
editing issue tracking profiles 68, 309
editing source control profiles 83, 333
issue tracking profiles 67, 308
source control profiles 82, 333
starting
execution plan 36, 250
Manual Testing Client 259
status
calculating 253
status calculation
execution plans 229
step properties
creating 22, 57
custom 57
deleting 57
editing 57
libraries 187
manual tests 187
page 58
storing
test results from Manual Testing Client 265
sub-reports
adding to reports 26, 387
deleting from reports 387
overview 387
subversion
adding source control profiles 87, 334
Subversion
editing source control profiles 88, 335
source control profiles 87, 334
success conditions
editing 145
tests 144
SVN
adding source control profiles 87, 334
editing source control profiles 88, 335
source control profiles 87, 334
switching
recent projects 46
switching modes
test coverage 113
synchronizing
data sources 98
internal and external issue states 298
requirements 122
requirements across tools 124
synchronizing requirements
based on schedules 124
system diagnostics
system diagnostics page 423
viewing 424
system requirements
SAP Solution Manager 314

T

Team Foundation Server
adding issue tracking profiles 74, 311
editing issue tracking profiles 75, 311
installing proxy service 74, 310
issue tracking profiles 74, 310
source control profiles 94, 326
Telelogic Doors
integration 61
Telelogic DOORS
configuring projects for integration 128, 320
installing on front-end server 128, 320
integrating 128, 320
test assignment
overview 212
test attachments
deleting 180
viewing 180
test containers
adding 145
adding links 146
editing 147

filtering 136
overview 145
test coverage
modes 113
switching modes 113
test coverage page
requirements 113
test coverage status
requirements 113
test elements
copy to other projects 135
copying 135
copying between projects 135
cutting 135
deleting 135
pasting 135
reordering 136
shortcuts 139
test export
Excel 198
test folders
adding 147
editing 147
filtering 136
overview 147
sorting 148
test import
Excel 198
Test Manager
installing 12
licensing 12
logging in 12
logging out 13
login 12
logout 13
welcome 6
test mapping file
editing 203
test packages
cleaning up 150
copying 150
creating 37, 148
External IDs 148
overview 148
reusing 150
reverting to test 150
test parameters
SilkTest Workbench 178, 352
test properties
configuring 156
finding 156
replacing 157
test reports
baseline comparison 142, 381
manual tests 142, 380
progress 141, 380
run comparison 142, 381
status 141, 379
test results
analyzing 295
test runs
analyzing 269
deleting results 270
details 271
viewing details 36, 269
test selection
overview 206
test status
changing 269
test steps
assigning issues to 171
calling 188
testing cycle capacity 209
testing cycles
assigning tests 210
creating 208
deleting 211
finishing 211
moving tests to other testing cycle 212
properties page 227
removing tests 213
removing tests from manual testers 214
starting 211
testing cycles properties page 227
TestPartner
adding source control profiles 79, 336
configuring 158, 354
editing source control profiles 80, 337
source control profiles 79, 336
tests
adding parameters 39, 175
assigning attributes 38, 173
assigning from grid view to execution plans 33, 242
assigning requirements 40, 178
assigning to execution plans through filter 34, 242
assigning to requirements from grid view 109
attaching files 40, 179
attaching links 41, 180
attachments page 180
attributes 173
baseline comparison report 142, 381
changes 151
changing run status 269
cleanup 231
contents 138
creating 37, 155
deleting attachments 180
deleting attributes 174
document view 134
editing 37, 155
editing attributes 174
enabling screen capturing 296
enabling video capturing 295
executing 250
executing trial runs 155
exporting to Excel 200
filtering in grid view 167
generating from requirements 29, 31, 110
grid view 165
grouping in grid view 166
history 151
history page 151
import and update 199
importing from excel 200

importing from Excel 200
integration default nodes 136
issues page 182
manual test reports 142, 380
manually assigning to execution plans 33, 242
manually assigning to requirements 110
multi-edit attributes 166
multi-edit properties 166
overview 134, 155
parameters 174
progress reports 141, 380
properties page 136
removing assigned requirements 179
reports 141, 379
run comparison reports 142, 381
run details 271
runs page 181
screen captures 295
setup 231
status reports 141, 379
steps 169, 190
success conditions 144
test containers 145
test steps 169, 190
toolbar functions 139
tracking history 151
tree 134
updating from Excel 200
video captures 295
viewing assigned executions 181
viewing attachments 180
viewing recent changes 151
tests tree
collapsing 135
expanding 135
text export
rules 199
text import
rules 199
text update
rules 199
TFS
adding issue tracking profiles 74, 311
adding source control profiles 94, 326
editing issue tracking profiles 75, 311
editing source control profiles 94, 327
installing proxy service 74, 310
issue tracking profiles 74, 310
source control profiles 94, 326
toolbar
execution planning 205
requirements 102
tests 139
toolbar functions
reports 392
tracking
overview 284
tracking history
requirements 114
tests 151
trial runs
tests 155

U

UNC
adding source control profiles 89, 338
editing source control profiles 90, 338
source control profiles 89, 337
uninstalling
Manual Testing Client 260
Universal Naming Convention
source control profiles 89, 337
updating
build informations 44
tests from Excel 200
updating from Word
requirements 120
updating requirements from Word
overview 120
Upload Manager
command-line options 153
downloading 152
installing 152
overview 152
uploading files to file pool 152
uploading
report templates 387
SilkPerformer test results 282, 347
test results from Manual Testing Client 264
uploading attachments
Manual Testing Client 263
uploading Excel files
data sources 98
uploading execution packages
managing change conflicts 264
uploading files
SilkCentral file pool 152
user interface
tour 6
using 6
using
user interface 6

V

Version Manager
adding source control profiles 80, 330
editing source control profiles 81, 330
source control profiles 80, 330
versioning
tests during Excel import 203
tests during Excel update 203
versions
comparing 19, 193
creating 19, 192
differences 19, 193
differing 19, 193
reverting 19, 193
viewing 193
VFS
adding source control profiles 91, 323
editing source control profiles 92, 324
source control profiles 90, 322
video capturing

enabling for tests 295
Manual Testing Client 269
viewing
assigned executions 181
external properties 132
requirement attachments 108
test attachments 180
test run details 36, 269
viewing execution result files
visual tests 352
viewing recent changes
requirements 114
tests 151
virtual configurations
VMware Lab Manager 284, 313
Virtual File System
source control profiles 90, 322
virtualizing
lab environments 284, 313
Visual SourceSafe
adding source control profiles 85, 328
source control profiles 85, 327
visual tests
viewing execution result files 352
VMware Lab Manager
integration 284, 313

W

Windows Script Host
log information 195, 357
parameters 195, 357
properties 195, 356
result files 195, 357
supported script languages 194, 356
viewing returned information 195, 357
Windows Script Host tests
overview 194, 355
Windows scripting
configuring 163, 355
Word
importing requirements 119
Word file
requirements 120
working offline
Manual Testing Client 266
working with tests
requirements 109
WSH
overview 194, 355
parameters 195, 357
properties 195, 356
supported script languages 194, 356
viewing returned information 195, 357